Data Discovery and Anomaly Detection Using Atypicality

A DISSERTATION SUBMITTED TO THE GRADUATE DIVISION OF THE UNIVERSITY OF HAWAIʻI AT MĀNOA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

IN

ELECTRICAL ENGINEERING

September 2017

By

Elyas Sabeti

Dissertation Committee:

Anders Høst-Madsen, Chairperson

Anthony Kuh

Narayana Prasad Santhanam

June Zhang

Kyungim Baek

To my parents, my wife and my lovely sister

Acknowledgements

First of all, I would like to express my sincere appreciation to my adviser, Prof. Anders Høst-Madsen for his support during these past five years. I am grateful for his constructive comments, productive criticisms and never-ending support through my Ph.D. journey. Thank you Prof. Høst-Madsen for all your time and help.

I would like to thank my committee members Prof. Anthony Kuh, Prof. Narayana Prasad Santhanam, Prof. Kyungim Baek and Prof. June Zhang for their interest in my work and their insightful comments. Special thanks to Prof. Todd R. Reed who did not deprive me of his helpful comments, even after his retirement.

I would like to acknowledge my parents, Maliheh and Alireza, for their support in every aspect of my life. Without their dedication, my Ph.D. journey could not have been accomplished. I also like to give my special thanks to my caring, loving and supportive wife Shabnam for all her encouragement and support since last year that she came to my life. Additionally, I'm thankful to my lovely younger sister Anita who is always my moral supporter.

Finally, I would like to thank my fellow labmates of many years: Nasir, Navid, Meysam and Maryam. We had a great time together and it almost felt like home. Also I'm grateful of my friends Poya, Saeed and Ehsan who accompanied me through this journey.

This work was supported in part by NSF grants CCF 1017823, 1017775, and 1434600 and the NSF Center for Science of Information (CSoI), an NSF Science and Technology Center, under grant agreement CCF-0939370.

Abstract

One characteristic of modern era is the exponential growth of information, and the ready availability of this information through networks, including the Internet – "Big Data." The question is what to do with this enormous amount of information. One possibility is to characterize it through statistics – think averages. The perspective of our approach is the opposite, namely that most of the value in the information is in the parts that deviate from the average, that are unusual, *atypical*. Think of art: the valuable paintings or writings are those that deviate from the norms, that are atypical. The same could be true for venture development and scientific research. The aim of atypicality is to extract small, rare, unusual and interesting pieces out of big data, which complements statistics about typical data.

We define atypicality as follows: a sequence is atypical if it can be described (coded) with fewer bits in itself rather than using the (optimum) code for typical sequences. This definition is based on the ability of a universal source coder (atypical encoder) to encode a sequence (or a subsequence) in fewer bits in comparison to the optimum encoder of typical data (typical encoder). An inaccurate typical model raises atypicality flag for all the sequences, and a naive atypical encoder can never catch an atypical sequence. So for a sequence, the difference between performance of an optimum typical encoder and a universal atypical encoders shows how atypical that sequence is. We measure the performance of an encoder by the number of bits it requires to describe a sequence. Thus atypicality can also be deduced as measure that depends on the performance of the optimum coder and a universal coder in describing a sequence. This is closely related to Rissanen's Minimum Descriptive Length (MDL).

In this work after defining the notion of atyicality, we first setup a framework for binary model to analyze our atypicality measure and verify its properties, then we extend our approach

to real-valued models. In our approach for real-valued models, we start with introducing two new predictive encoders for accurate description length called Normalized Likelihood Method (NLM) and Sufficient Statistic Method (SSM) which improves the redundancy of predictive MDL, then we use asymptotic MDL to derive analytical results. Our algorithms has been applied to various sources of Big Data such as heart rate Holter monitoring, DNA, 15 years of stock market and 2 years of oceanographic data to find arrhythmias, viral and bacterial infections, unusual stock market behavior and whale vocalization, respectively.

Contents

List of Figures

List of Tables

1

Introduction

One characteristic of the information age is the exponential growth of information, and the ready availability of this information through networks, including the Internet – "Big Data." The question is what to do with this enormous amount of information. One possibility is to characterize it through statistics – think averages. The perspective in our method is the opposite, namely that most of the value in the information is in the parts that deviate from the average, that are unusual, atypical. The rest is just background noise.

Atypical data can be thought of as anomalies [1]. However, in the era of "Big Data," such anomalies takes on a new relevance and a broader meaning than perhaps in traditional anomaly detection. There is a need to make sense out of huge amounts of data that we

perhaps do not know too much about. Since we cannot examine all of the data, it makes sense to try to extract a few data points that are "interesting." However, we might not know in advance what makes data interesting. We are looking for "unknown unknowns" [2]. Instead of looking at specific statistics of data, we need to use a universal approach. We believe the most relevant framework for universality is information theory, specifically source coding. Prior work on using universal source coding for anomaly detection [1, 3–12] have principally been heuristic. Our methodology is developing a general framework for universal data discovery based on theoretical analysis. This framework then can be implemented in various ways.

1.1 Applications

Atypicality is relevant in large number of various applications. Atypicality is related to other methodologies such as anomaly detection [1, 13–18] and quickest change detection [19–32] and data mining [33], yet it has a unique flavor in the sense that we are not looking for something specific. We will list a few applications here.

ECG. For electrocardiogram (ECG) recordings there are patterns in heart rate variability that are known to indicate possible heart disease [34–37]. With modern technology it is possible for an individual to wear and unobtrusive heart rate monitor 24/7. If atypical patterns occur, it could be indicative of disease, and the individual or a doctor could be notified. But perhaps a more important application is to medical research. One can analyze a large collection of ECG recordings and look for individuals with atypical patterns. This can then potentially be used to develop new diagnostic tools.

Genomics. Another example of application is interpretation of large collections of genomics data. Given that all mammals have essentially the same set of genes, there must exist some significant differences that distinguish the obvious distinct attributes between species, as well as more subtle differences within a species. Although the genome has been mined by exhaus-

tive studies applying a panoply of approaches, regions once thought to be "uninteresting" have recently come under increased study for their potential role in defined morphological and physiological differences between individuals [38]. Applying an atypical evaluation tool to genomic data from individuals of known pathophysiological/morphological irregularities may provide valuable insight to the genetic mechanisms underlying the condition.

Ocean Monitoring. In passive acoustic monitoring (PAM) of oceans, one or more hydrophones is towed behind a ship or deployed in a fixed bottom-mounted or suspended array in order to record vocalizations of marine mammals. One major focus is to detect, and perhaps count, rare or endangered species. It would be highly interesting to scan the data for any unusual patterns, which can then be further examined by a researcher.

Plant Monitoring. In for example nuclear plants, atypical monitoring data may be indicative of something about to go wrong.

Computer Networks. Atypical network traffic could be indicative of a cyberattack. This is already being used through anomaly detection [13]. However, an abstract atypicality approach can be used to find more subtle attacks – the unknown unknowns.

Airport Security. Already software is being used to flag suspicious flyers, likely based on past attacks. Atypical detection could be used to find innovative attackers.

Stock Market. Atypicality could be used to detect insider trading. It could also be used by investors to find unusual stocks to invest in, promising outstanding returns – or ruin.

Electric Power Grids. In the ten year period between 2003 and 2012 there were an estimated 679 widespread electrical power grid outages due to severe weather and estimates of annual costs were between $18B to $33B per year. Many of these failures were the result of cascading failures that could be found by quick detection of atypical electrical grid data.

Astronomy. Atypicality can be used to scan huge databases for new kinds of cosmological phenomena.

Credit Card Fraud. Unusual spending patterns could be indicative of fraud. This is

already used by credit card companies, but obviously in a simple, and annoying way, as anyone who's credit card has been blocked on an overseas trip can testify to.

Gambling. Casinos are constantly fighting fraudsters. This is a game of cat and mouse. Fraudsters constantly find new ways to trick the casinos (one such inventor was Shannon himself). Therefore, an abstract atypicality approach may be the best solution to catch new ways of fraud.

1.2 Notation

We use x to denote a sequence in general, x_n or $x[n]$ denotes a single sample of the sequence. Also x^n or x_1^n denotes the sequence $x_1, x_2, \ldots, x_n$; we use $\mathbf{x}^n$ to represent a sequence of n vectors. We use capital letters X_i to denote random variables rather than specific outcomes. All logarithms are to base 2 unless otherwise indicated.

1.3 Source Coding

Since readers of this text might have different backgrounds, a brief introduction on source coding is needed. All the contents of this section are derived from [39] and readers are encouraged to refer to this book for more detail.

1.3.1 Class of Codes

Suppose C is a source code for a random variable X with the support $\mathcal{X}$, then C is a mapping from $\mathcal{X}$ to the set of finite-length strings of symbols $\mathcal{D}^*$ with a alphabet size of D (encoding), and similarly C^{-1} is a mapping from the set of finite-length strings of symbols $\mathcal{D}^*$ to source alphabet $\mathcal{X}$ (decoding). As an example through this section, consider the four nitrogen bases found in DNA: Adenine, Cytosine, Guanine, and Thymine as outcome of a random variable

X, i.e., $\mathcal{X} = \{A, C, G, T\}$ and suppose $\mathcal{D} = \{0, 1\}$. A source code for these DNA bases can be $C(A) = 0$, $C(C) = 1$, $C(G) = 01$ and $C(T) = 10$. Another source code can be $C(A) = 00$, $C(C) = 01$, $C(G) = 10$ and $C(T) = 11$, and many other source codes can be assigned to these bases, but which one is the best? In order to answer this question, we should first define what is considered as a "good" source code? These questions can be answered after defining the *expected length*.

The expected length $L(C)$ of a source code $C(x)$ for a random variable X with the support $\mathcal{X}$ and the probability mass function $P(x)$ is give by

$$L(C) = \sum_{x \in \mathcal{X}} P(x) l(X)$$

where $l(x)$ is the length of the codeword assigned to each $x \in \mathcal{X}$. Now a desirable source code is the one with minimum expected length. But is there any constraint in this minimization problem? Suppose the DNA base source is uniform ($P(A) = P(C) = P(G) = P(T) = \frac{1}{4}$) and consider the following source coder that assigns only one bit to each source symbols $C(A) = 0$, $C(C) = 0$, $C(G) = 0$ and $C(T) = 0$ (Singular code). Obviously this source code minimizes the expected length, but the decoder cannot decode the received sequences uniquely. In fact a "good" source coder solve an optimization problem: we would like to choose a class of source codes that minimizes the expected length $L(C)$ while being able to the uniquely decode the encoded sequence of bits at the decoder without waiting for the upcoming bit stream. Therefore singular code is not a choice.

Now assume the following source code $C(A) = 0$, $C(C) = 010$, $C(G) = 01$ and $C(T) = 10$. Obviously this code is not singular, but still it's not uniquely decodable since $C(GA) = C(C) = 010$. So consider the source code $C(A) = 10$, $C(C) = 00$, $C(G) = 11$ and $C(T) = 110$. This source code is uniquely decodable, but it's not instantaneous, i.e., when the decoder receives 11, it should wait for the next symbol to choose between G and T since the code for G is a *prefix* for T code. Finally the source code $C(A) = 0$, $C(C) = 10$, $C(G) = 110$ and $C(T) = 111$ is uniquely decodable and instantaneous, this class of codes are called *instantaneous* or prefix code.

Even though in the aforementioned DNA bases source examples, the alphabet of the source and the code strings are different, it is not a required assumption in many practical applications. For instance, whenever you compress a file in a computer, the process is a mapping from binary source to a binary code with memory reduction goals.

1.3.2 Optimum Source Coder

For any prefix code over a D-ary alphabet, the codeword lengths $l_1, l_2, ..., l_m$ satisfy the inequality

$$\sum_i D^{-l_i} \leq 1$$

This is called *Kraft inequality*. Conversely, the Kraft inequality is a sufficient condition for the existence of a prefix code. Now we consider the problem of finding the prefix code with the minimum expected length. Suppose the codeword lengths $l_1, l_2, ..., l_m$ satisfy the Kraft inequality, we have to minimize the expected codelength

$$L = \sum_i p_i l_i$$

by solving with optimization problem we obtain

$$p_i = D^{-l_i} \qquad l_i^* = -\log_D p_i \qquad L^* = \sum p_i l_i^* = -\sum p_i \log_D p_i = H_D(X)$$

where $H_D(X)$ is the *entropy* of the source. Since l_i should always be an integer, and $l_i^* = -\log_D p_i$ is not always an integer, the expected codelength of an optimum source coder is within 1 bit of the entropy of the source

$$H_D(X) \leq L < H_D(X) + 1$$

Hence the entropy of a source is the fundamental lower limit of the expected codelength. As a result, when the probability mass distribution $P(x)$ of a source is known, the optimal codelength of each source symbol is $l(x) = \lceil -\log_D P(x) \rceil$. Famous optimum source codes are Huffman code and Shannon-Fano-Elias code, for more information see [39].

1.3.3 Universal Source Coder

For an optimum encoder, we assumed that the probability mass function of the source code is known, however in many practical situations, this is not the case. Instead all we have is the sequence that need to be encoded. In these cases, universal source coders are used. There are many innovative and intuitive methods for universal source coders in the literature, but here we only introduce a simple one to give an idea to readers about how these algorithms work. Again more methods are covered in [39].

Let's consider binary sequence $x^n \in \{0,1\}^n$ of length n that we need to encode. After seeing the entire sequence, first we count the number of 1's in the sequence, i.e., $k = \sum_{i=1}^{n} x_i$ and encode the integer k using $\lceil \log n \rceil$ bits. Then we have to encode the index of this sequence among all sequences that have k 1's using $\lceil \log \binom{n}{k} \rceil$ bits. Now using Wozencraft-Reiffen lemma we have $\log \binom{n}{k} \leq nH(\frac{k}{n}) - \frac{1}{2}\log n$ for $k \neq 0$ and $k \neq n$. Therefore the total codelength is

$$l(x^n) \leq nH(\frac{k}{n}) + \frac{1}{2}\log n$$

for $k \neq 0$ and $k \neq n$. It means that the cost of encoding the sequence is about $\frac{1}{2}\log n$ above the optimal codelength if Shannon-Fano-Elias code being used for Bernoulli distribution corresponding to $P(1) = \frac{k}{n}$.

The problem with this type of encoding is that, the encoder have to wait until the entire sequence is seen. Instead, methods using mixture distribution can be used that can encode the sequence on the fly. In fact, if we choose a uniform mixture of all Bernoulli distributions on x^n, the same codelength can be achieved. Another caveat with these offline approach and

online uniform mixture approach is that, the codelength bound does not apply for $k = 0$ and $k = n$. In online approach, if instead of uniform distribution, we use Dirichlet$(\frac{1}{2}, \frac{1}{2})$ distribution, the total codelength will be

$$l(x^n) \leq nH(\frac{k}{n}) + \frac{1}{2}\log n + \log\frac{\pi}{8}$$

that is also valid for $k = 0$ and $k = n$.

There are more sophisticated approaches such as arithmetic coding, Lempel-Ziv coding and Context Tree coder that we don't go over here, but interested readers can find more details in [39].

2

Atypicality

2.1 Atypicality Definition

Our starting point is the in theory of randomness developed by Kolmogorov and Martin-Löf [39–41]. Kolmogorov divides (infinite) sequences into 'typical' and 'special.' The typical sequences are those that we can call random, that is, they satisfy all laws of probability. They can be characterized through Kolmogorov complexity. Kolmogorov complexity has an abstract definition: the Kolmogorov complexity of an object is the length of the shortest computer program (in a universal computer) that produces the object as output. Now a sequence of bits $\{x_n, n = 1, \ldots, \infty\}$ is random (i.e, iid uniform) if the Kolmogorov complexity

of the sequence satisfies $K(x_1, \ldots, x_n) \geq n - c$ for some constant c and for all n [40]. The sequence is incompressible if $K(x_1, \ldots, x_n|n) \geq n$ for all n, and a finite sequence is algorithmically random if $K(x_1, \ldots, x_n|n) \geq n$ [39]. In terms of coding, an iid random sequence is also incompressible, or, put another way, the best coder is the identity function. Let us assume we draw sequences x^n from an iid uniform distribution. The optimum coder is the identity function, and the code length is n. Now suppose that for one of these sequences we can find a (universal) coder so that the code length is less than n; while not directly equivalent, one could state this as $K(x_1, \ldots, x_n|n) < n$. With an interpretation of Kolmogorov's terms, this would not be a 'typical' sequence, but a 'special' sequence. We will instead call such sequences 'atypical.' Considering general distributions and general (finite) alphabets instead of iid uniform distributions, we can state this in the following general principle

Definition 1. A sequence is atypical if it can be described (coded) with fewer bits in itself rather than using the (optimum) code for typical sequences. □

This definition is central to our approach to the atypicality problem.

In the definition, the "(optimum) code for typical sequences," is quite specific, following the principles in for example [39]. We assume prefix free codes. Within that class the coding could be done using Huffman codes, Shannon codes, Shannon-Fano-Elias codes, arithmetic coding etc. We care only about the code length, and among these the variation in length is within a few bits, so that the code length for typical encoding can be quite accurately calculated.

On the other hand, "described (coded) with fewer bits in itself" is less precise. In principle one could use Kolmogorov complexity, but Kolmogorov complexity is not calculable and it is only given except for a constant, and comparison with code length therefore is not an "apples-to-apples" comparison. Rather, some type of universal source coder should be used. This can be given a quite precise meaning in the class of finite state machine sources, [42] and following work, and is strongly related to minimum description length (MDL) [42–45]. What

is essential is that we adhere to *strict decodability* at the decoder. The decoder only sees a stream of bits, and from this it should be able to accurately reconstruct the source sequence. So, for example, if a sequence is atypical, there must be a type of "header" telling the decoder to use a universal decoder rather than the typical decoder. Or, if atypical sequences can be encoded in multiple ways, the decoder must be informed through the sequence of bits which encoder was used. One could argue that such things are irrelevant for for example anomaly detection, since we are not actually encoding sequences. The problem is that if such terms are omitted, it is far too easy to encode a sequence "in itself." This is like choosing a more complex model to fit data, without accounting for the model complexity in itself, which is exactly what MDL sets out to solve, although also in this case actual encoding is not done. We therefore try to account for all factors needed to describe data, and we believe this is one of the key strengths of the approach.

There is a fine distinction between a sequence being atypical and anomalous, perhaps merely in semantics. Usually, we think of an anomaly as something caused by an outside phenomenon: an intruder in a computer network, a heart failure, a gambler playing tricks. If our detector fails to give an indication of such anomalies, we have a miss (or type II error), but if it gives an indication when such things are not happening we have a false alarm (type I error). Atypicality, on the other hand, is purely a property of data. Ideally, there are therefore no misses or false alarms: data is atypical or not. Here is what we mean. If there is an anomaly that expresses itself through the observed data, that must mean that there is some structure in the data, and in theory a source coder would discover and exploit such structure and reduce code length. Thus, if the data is not atypical that means there is simply no way to detect the anomality through the observations – again in theory. We therefore cannot really call that a miss. On the other hand, suppose that in a casino a gambler has a long sequence of wins. This could be due to fraud, but it could also be simply due to randomness. But casino security would be interested in either case for further scrutiny. Thus, the reason for the atypicality does not really matter, the atypicality itself matters. Still, to distinguish the two cases we call a sequence *intrinsically* atypical if it is atypical according to Definition

1 while being generated from the typical probability model, while it is *extrinsically* atypical if it is in fact generated by any other probability law.

Definition 1 has two parts that work in concert, and we can write it simplified as $C_t(x) - C_a(x) > 0$. The typical code length $C_t(x)$ is simply an expression of the likelihood of seeing a particular sequence. If $C_t(x)$ is large it means that the given sequence is unlikely to happen, and detecting sequences by $C_t(x) > \tau$ would catch many outliers. As an extreme example, if a sequence is impossible according to the typical distribution, $C_t(x) = \infty$, and it would always be caught. But it would not work universally. If, as we started out with, typical sequences are iid uniform, any sequence is equally likely and $C_t(x) > \tau$ would not catch any sequences. In this case, if a test sequence has some structure, it is possible that $C_a(x) < C_t(x)$, and such sequences would be caught by atypicality; thus calculating $C_a(x)$ is essential. Calculating $C_t(x)$ is also essential. Suppose that we instead use $E[C_t] - C_a(x)$, where $E[C_t]$ is the code length used to encode typical sequences "on average." Again, this will catch some sequences: if a test sequence has more or less structure than typical sequences, $E[C_t] - C_a(x) \neq 0$. But again, it will omit very obvious examples: if as test sequence we use a typical sequence with 0 and 1 swapped, $E[C_t] \approx C_a(x)$, while on the other hand $C_t(x) > C_a(x)$. And impossible sequences with $C_t(x) = \infty$ would not be caught with absolute certainty. Now, to declare something an outlier, we have to found a coder with $C_a(x) < C_t(x)$. It is not sufficient that $C_t(x)$ is large, i.e., that the sequence is unlikely to happen. However, we can always use the trivial coder that transmits data uncoded. If the sequence is unlikely to happen according to the typical distribution, then it is likely that $C_t(x) >$ (length of x).

Thus, it can be seen that the two parts work in concert to catch sequences. Each part might catch some sequences, but to catch all "anomalies," both parts have to be used.

Another point of view is the following. Suppose again the typical model is binary uniform iid. We look at a collection of sequences, and now we want to find the *most* atypical sequences, i.e., the most "interesting" sequences. Without a specification of what "interesting" is, it seems reasonable to choose those sequences that have the most structure, and again this

can reasonably be measured by how much the sequence can be compressed. This is what Rissanen [42] calls "useful information," $U(x) = n - C_a(x)$. But again, we need to take into account the typical model if it is not uniform iid. For example, if typical sequences have much structure, then sequence with little structure might be more interesting. We therefore end up with that $C_t(x) - C_a(x)$ is a reasonable measure of how interesting sequences might be. A caveat is that model inaccuracies can distort this measure.

2.2 Atypicality and Alternative Approaches Using Source Coding

Information theory and universal source coding has been used previously in anomaly detection, e.g., [1, 3–12, 31]. Even though a literature review of these methods and comparison with atypicality is essential, I will postpone it to the last chapter, since without having a full knowledge of the brand new notion of atypicality, any comparison is hard to understand. In the last chapter we'll see that if any anomaly detection method based on universal source coding detects an anomaly, atypicality will detect it too; however, atypicality can find anomalies that are undetectable to other approaches.

3

Discrete Case

Let us first consider the binary IID case, then we'll generalized it to non-IID case.

3.1 Binary IID Case

In order to clarify ideas, at first we consider a very simple model. The typical model is iid binary with $P(X_n = 1) = p$. The 'anomalies' are also binary iid but with $P(X_n = 1) = \theta$, where θ is unknown. We want to decide if a given sequence x^l is typical or atypical. This

can be stated as the hypothesis test problem

$$H_0 : \theta = p$$
$$H_1 : \theta \neq p$$

This problem does not have an UMP (universal most powerful) test. However, a common approach to solving this type of problem is the GLRT (generalized likelihood ratio test) [46]. Let

$$P(b) = P(X_n = b)$$
$$\hat{P}(b) = \frac{N(b|x^l)}{l}$$

where l is the sequence length and $N(b|x^l)$ is the number of $x_n = b \in \{0, 1\}$. Now the GLRT is

$$\begin{aligned} L &= \log \frac{\prod_{b=0}^{1} \hat{P}(b)^{N(b|x^l)}}{\prod_{b=0}^{1} P(b)^{N(b|x^l)}} \\ &= \sum_{b=0}^{1} N(b|x^l) \log \frac{1}{l} N(b|x^l) - \sum_{b=0}^{1} N(b|x^l) \log P(b) \\ &= l \sum_{b=0}^{1} \hat{P}(b) \log \frac{1}{l} N(b|x^l) - l \sum_{b=0}^{1} \hat{P}(b) \log P(b) \\ &= lD(\hat{p}\|p) \end{aligned}$$

$$\phi(x^l) = \begin{cases} 1 & L > t \\ 0 & L \leq t \end{cases} \tag{3.1}$$

Where $D(\hat{p}\|p) = \sum_{b=0}^{1} \hat{P}(b) \log \frac{\hat{P}(b)}{P(b)}$ is the relative entropy [39] and t some threshold. While the GLRT is a heuristic principle, it satisfies some optimality properties, and in this case it is equal to the invariant UMP test [47], which can be considered an optimum solution under certain constraints. Thus, it is reasonably to take this as the optimum solution for

this problem, and we do not need to appeal to Kolmogorov or information theory to solve the problem.

The complications start if we consider sequences of variable length l. The test (3.1) depends on the sequence length. We need to choose a threshold $t(l)$ as a function of l, which will then result in a false alarm probability $P_{FA}(t(l))$ and detection probability $P_D(t(l))$. There is no obvious argument for how to choose $t(l)$ from a hypothesis testing point of view; we could choose t independent of l, but that is just another arbitrary choice.

We will consider this problem in the context of Definition 1. In order to do so, we need to model the problem from a coding point of view. We assume we have an (infinite) sequence of sequences of variable length l_i, and these need to be encoded. We need to encode each bit, and also to encode whenever a new sequence starts. For typical encoding of the bits we can use a Shannon code, Huffman code, arithmetic coding etc. The code length for a sequence of length l is

$$\begin{aligned} L_t &= N(1|x^l)\log\frac{1}{p} + N(0|x^l)\log\frac{1}{1-p} \\ &= l\left(\hat{p}\log\frac{1}{p} + (1-\hat{p})\log\frac{1}{1-p}\right) \end{aligned} \tag{3.2}$$

except for a small constant factor; here $\hat{p} = \frac{1}{l}\sum x_i$. We also need to encode where a sequence ends and a new one starts. For simplicity let us for now assume lengths are geometrically distributed. We can then model the problem as one with three source symbols '0', '1' and ',' with an iid distribution with $P(',') = \epsilon$, $P('0') = p - \frac{\epsilon}{2}$, $P('1') = (1-p) - \frac{\epsilon}{2}$. If we assume ϵ is small, the expression (3.2) is still valid for the content part, and to each sequence is added a constant $-\log\epsilon$ to encode separators. To decide if a sequence is atypical according to Definition 1, we can use the universal source coder from [39]: the source encodes first the number of ones k; then it enumerates the sequences with k ones, and transmits the index of the given sequence. For analysis it is essential to have a simple expression for the code length. We can therefore use $L_a = lH(\hat{p}) + \frac{1}{2}\log l$, where $\hat{p} = \frac{1}{l}\sum x_i$. This is an approximation which is good for reasonably large l and it also reaches the lower bound in

[42, 48]. The source-coder also needs to inform the decoder that the following is an atypical sequence (so that it knows to use the atypical decoder rather than the typical encoder), and where it ends. For the former we can use a '.' to indicate the start of an atypical sequence rather than the ',' for typical sequences. If the probability that a sequence is atypical is $\delta \ll 1$, $P('.') = \delta\epsilon$ and $P(',') = (1-\delta)\epsilon \approx \epsilon$. The code length for a '.' now is $-\log\epsilon - \log\delta$. To mark the end of the atypical sequence we could again insert a '.' or a ','. But the code for either is based on the distribution of lengths of *typical* sequences, which we assume known, whereas we would have no knowledge of the length of atypical sequences. Instead it seems more reasonable to encode the length of the specific atypical sequence. As argued in [43, 49] this can be done with $\log^* l + \log c$, where c is a constant and

$$\log^*(l) = \log l + \log\log l + \log\log\log l + \cdots \tag{3.3}$$

where the sum continues as long as the argument to the log is positive. To summarize we have

$$\begin{aligned} L_t &= l\left(\hat{p}\log\frac{1}{p} + (1-\hat{p})\log\frac{1}{1-p}\right) - \log\epsilon \\ L_a &= lH(\hat{p}) + \frac{1}{2}\log l + \log^* l + \log c - \log\epsilon - \log\delta \\ &\approx lH(\hat{p}) + \frac{3}{2}\log l - \log\epsilon + \tau \\ \tau &= -\log\delta + \log c \end{aligned} \tag{3.4}$$

The criterion for a sequence to be atypical is $L_a < L_t$, which easily seen to be equivalent to

$$D(\hat{p}\|p) > \frac{\tau + \frac{3}{2}\log l}{l} \tag{3.5}$$

If the lengths are fixed, this reduces to (3.1). But if the lengths are variable, (3.5) provides a threshold as a function of l. The term $\frac{3}{2}\log l$ ensures that $\lim_{l\to\infty} P_{FA}(l) = 0$, which seems reasonable. If instead $D(\hat{p}\|p) > \frac{\tau}{l}$ is used, it is easy to see that $\lim_{l\to\infty} P_{FA}(l) > 0$. Except for this property, the term $\frac{3}{2}\log l$ might seem arbitrary, e.g., why $\frac{3}{2}$? But it is based on solid

theory, and as will be seen later it has several important theoretical properties.

We will examine the criterion (3.5) in more detail. The inequality (3.5) gives two thresholds for $\hat{p}$,

$$\hat{p} > p_+$$
$$\hat{p} < p_-$$

Where $0 < p_- < p < p_+ < 1$. It is impossible to find explicit expressions for $p_\pm$, but it is clear that

$$p_\pm \to p \quad \text{as } l \to \infty.$$

Therefore, for l large, we can replace $D(\hat{p}\|p)$ with a series expansion. We then end up with the more explicit criterion

$$\begin{aligned} \frac{(p-\hat{p})^2}{pq\ln 4} &> \frac{1}{l}(\tau + \frac{3}{2}\log l) \\ |\hat{p}-p| &> \Delta\tau \doteq \sqrt{\frac{pq\ln 4}{l}}\sqrt{\tau + \frac{3}{2}\log l} \end{aligned} \tag{3.6}$$

In the following we will use this as it is considerably simpler to analyze. We can also write this as

$$\left| \frac{\sum_{i=1}^{l} x_i - p}{\sqrt{pql}} \right| > \sqrt{2\tau \ln 2 + 3\ln l} \tag{3.7}$$

Now, if not for the term $3\ln l$, this would be a central limit type of statement, and the probability that a sequence is classified as (intrinsically) atypical would be

$$P_A \approx 2Q\left(\sqrt{2\ln 2\tau}\right) \tag{3.8}$$

independent of l. Or main interest is exactly the the dependency on l, which is given by the

following Theorem

Theorem 1. Consider an iid $\{0,1\}$-sequence. Let $P_A(l)$ be the probability that a sequence of length l is classified as *intrinsically* atypical according to (3.6). Then $P_A(l)$ is bounded by

$$\begin{aligned} P_A(l) \leq & 2^{-\tau+1}\frac{1}{l^{3/2}}K(l,\tau) \\ \forall\tau : \lim_{l\to\infty} K(l,\tau) = & 1 \end{aligned} \tag{3.9}$$

For $p = \frac{1}{2}$ this can be strengthened to

$$P_A(l) \leq 2^{-\tau+1}\frac{1}{l^{3/2}} \tag{3.10}$$

These bounds are tight in the sense that

$$\lim_{l\to\infty} \frac{\ln P_A(l)}{-\frac{3}{2}\ln l} = 1 \tag{3.11}$$

Proof The Chernoff bound (e.g., [50]) states

$$\begin{aligned} P_A(l) = P\left(|\hat{p}-p| > \Delta\tau\right) = & 2P\left(\sum_{i=1}^{l} X_i \geq lp + b\right) \\ \leq & 2\inf_{s>0}\left\{e^{-lsp-sb}M_X(s)^l\right\} \end{aligned}$$

Where (as usual, $q = 1-p$)

$$b = \sqrt{lpq\ln 4}\sqrt{\tau + \frac{3}{2}\log l}$$

and $M_S(s)$ is the moment generating function of X_i, which for the binomial random variable is

$$M_X(s) = pe^s + q$$

Then

$$\frac{1}{2}P_A(l) \leq \inf_{s>0}\left\{\exp\left(-s\left(pl+b\right)\right)\left(pe^s+q\right)^l\right\}$$

Minimizing over s gives

$$\frac{1}{2}P_A(l) \leq \left(\frac{lq}{lq-b}\right)^l\left(\frac{q(lp+b)}{p(lq-b)}\right)^{-lp-b}$$

or

$$\begin{aligned}
\ln\frac{1}{2}P_A(l) \leq & l\ln\left(\frac{lq}{lq-b}\right)+(-lp-b)\ln\left(\frac{q(lp+b)}{p(lq-b)}\right)\\
= & l\ln\left(1+\frac{b}{lq-b}\right)+(-lp-b)\ln\left(1+\frac{b}{p(lq-b)}\right)\\
\leq & \frac{b^2\left(3l^2q^2p+lb(7p^2-6p-3)+b^2(6p+3)\right)}{6p^2(b-lq)^3}\\
\leq & -\frac{b^2}{2lpq}+O\left(1\right)\frac{b^3}{l^2}\\
= & -\tau\ln 2-\frac{3}{2}\ln l+O\left(1\right)\frac{\ln^{3/2}l}{\sqrt{l}}\tau^{3/2}, \qquad (3.12)
\end{aligned}$$

where we have used $x-\frac{x^2}{2}\leq\ln(1+x)\leq x-\frac{x^2}{2}+\frac{x^3}{3}$ for $x\geq 0$. The equation (3.12) directly leads to (3.9).

Using Hoeffding inequality we also get the bound

$$\begin{aligned}
P_A(l) &\leq 2\exp\left(-2\frac{b^2}{l}\right)\\
&= 2\exp\left(-4pq\ln 2\left(\tau+\frac{3}{2}\log l\right)\right)
\end{aligned}$$

for $p=\frac{1}{2}$ this is tighter than (3.12).

For the lower bound we use moderate deviations from [51]. Define $\tilde{X}_i=\frac{X_i-p}{pq}$. We can then

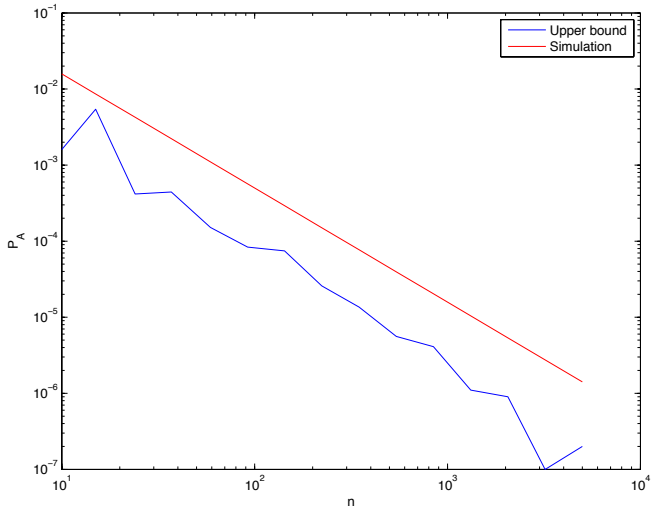

Figure 3.1: Simulated P_A and the Upper bound for $\tau = 1, p = 0.3$.

rewrite (3.7) as

$$\left| \frac{\sum_{i=1}^{l} \tilde{X}_i}{\sqrt{l(2\tau \ln 2 + 3 \ln l)}} \right| > 1$$

We define $a_l = \frac{1}{2\tau \ln 2 + 3 \ln l}$, which satisfies $\lim_{l\to\infty} a_l = 0$, $\lim_{l\to\infty} l a_l = \infty$. Using this as a_l in [51, Theorem 3.7.1] gives

$$\begin{aligned}
&\liminf_{l\to\infty} \frac{1}{2\tau \ln 2 + 3 \ln l} \ln P\left(\left| \frac{\sum_{i=1}^{l} \tilde{X}_i}{\sqrt{l(2\tau \ln 2 + 3 \ln l)}} \right| > 1 \right) \\
&= \liminf_{l\to\infty} \frac{1}{3 \ln l} \ln P\left(\left| \frac{\sum_{i=1}^{l} \tilde{X}_i}{\sqrt{l(2\tau \ln 2 + 3 \ln l)}} \right| > 1 \right) \\
&\geq -\frac{1}{2}
\end{aligned}$$

Together with the upper bound, this gives (3.11). □

Figure 3.1 compares the upper bound with simulations.

We can also bound the detection probability for *extrinsically* atypical sequences as follows

Theorem 2. Suppose that the typical sequence is iid $\{0, 1\}$-sequence with $P(X_n = 1) = p$.

Let the test sequence by iid with $P(X_n = 1) = p_a$. The probability that the test sequence is missed according to criterion (3.6) is upper bounded by

$$\begin{aligned} P_M(l) \leq & 2^{-\tau} \frac{1}{l^{3/2}} \left(\frac{q_a p}{p_a q} \right)^{\sqrt{lpq(2\tau \ln 2 + 3 \ln l)}} \\ & \times \left(\frac{q_a^{p-1} q^{p+1}}{p_a^p p^p} \right)^{-l} K(l, \tau) \end{aligned} \tag{3.13}$$

$$\forall \tau : \lim_{l \to \infty} K(l, \tau) = 1$$

Proof We may assume that $p_a < p$. Then the Chernoff bound states

$$\begin{aligned} P_M(l) = P\left(|\hat{p} - p| < \Delta\tau \right) \leq & P\left(\sum_{i=1}^{l} X_i \geq lp - b \right) \\ \leq & \inf_{s>0} \left\{ e^{-lsp + sb} M_X(s)^l \right\} \end{aligned}$$

Where (as usual, $q = 1 - p$)

$$b = \sqrt{lpq \ln 4} \sqrt{\tau + \frac{3}{2} \log l}$$

and $M_S(s)$ is the moment generating function of X_i, which for the binomial random variable is

$$M_X(s) = p_a e^s + q_a$$

Then

$$P_M(l) \leq \inf_{s>0} \left\{ \exp\left(-s\left(pl - b \right) \right) \left(p_a e^s + q_a \right)^l \right\}$$

Minimizing over s gives

$$P_M(l) \leq \left(\frac{lq_a}{lq+b}\right)^l \left(\frac{q_a(lp-b)}{p_a(lq+b)}\right)^{-lp+b}$$

or

$$\begin{aligned}\ln P_M(l) \leq & l \ln\left(\frac{lq_a}{lq+b}\right) + (-lp+b)\ln\left(\frac{q_a(lp-b)}{p_a(lq+b)}\right) \\ \leq & l\ln\left(\frac{q_a}{q}\right) - lp\ln\left(\frac{q_a}{p_a}\right) - lp\ln\left(\frac{q}{p}\right) \\ & + b\left(\ln\left(\frac{p}{q}\right) + \ln\left(\frac{q_a}{p_a}\right)\right) - \frac{b^2}{2lpq} + O\left(\frac{b^3}{l^2}\right)\end{aligned}$$

using series expansions. □

3.1.1 Hypothesis testing interpretation

The solution (3.5) may seem arbitrary, but it has a nice interpretation in terms of hypothesis testing [52]. Return to the solution (3.1). That solution gives a test for a given l. However, the problem is that it does not reconcile tests for different l. One way to solve that issue is to consider l a random variable, i.e., introducing a prior distribution in the Bayesian sense. Let the prior distribution of l be $P_L(l)$. The equation (3.1) now becomes

$$\begin{aligned}L &= \log \frac{\prod_{b=0}^1 \hat{P}(b)^{N(b|x^l)} P_L(l)}{\prod_{b=0}^1 P(b)^{N(b|x^l)} P_L(0)} \\ &= l\sum_{b=0}^1 \hat{P}(b)\log\frac{1}{l}N(b|x^l) \\ &- l\sum_{b=0}^1 \hat{P}(b)\log P(b) + \log P_L(l) - \log P_L(0) \\ &= lD(\hat{p}\|p) + \log P_L(l) - \log P_L(0)\end{aligned}$$

The hypothesis test now is

$$D(\hat{p}\|p) > \frac{\tau + \log P_L(0) - \log P_L(l)}{l} \tag{3.14}$$

Of course, the problem is that we don't know $P(l)$. Still, compare that with (3.5) without the approximations,

$$D(\hat{p}\|p) > \frac{\tau + \frac{1}{2}\log l + c + \log^* l}{l} \tag{3.15}$$

To the term $c + \log^* l$ corresponds a distribution on the integers, namely $Q(l)$ in [43, (3.6)]. Except for the term $\frac{1}{2}\log l$, the equations (3.14) and (3.15) are identical if we use the prior distribution $P_L(l) = Q(l)$. Rissanen [43] argues that the distribution $Q(l)$ is the most reasonable distribution on the integers when we have really no prior knowledge, mainly from a coding point of view. This therefore seems a reasonable distribution for $P(l)$. What about the term $\frac{1}{2}\log l$? The model for the non-null hypothesis has one unknown parameter, p, so that it is more complex than the null hypothesis. We have to account for this additional complexity. Our goal is to find an explanation for atypical sequences among a large class of explanations, not just the distribution of zeros and ones. If there is no "penalty" for finding a complex explanation, any data can be explained, and all data will be atypical. This is Occam's razor [39]. The "penalty" for one unknown parameter as argued by Rissanen is exactly $\frac{1}{2}\log l$. We therefore have the following explanation for (3.5),

> The criterion (3.5) can be understood as a hypothesis test with prior distribution $Q(l)$ [43] and penalty $\frac{1}{2}\log l$ for the unknown parameter.

Seen in this light, Theorem 1 is not surprising. In (3.5) we have replaced $\frac{1}{2}\log l + \log^* l$ with $\frac{3}{2}\log l$, which implicitly corresponds to the prior distribution $P_L(l) \sim l^{-3/2}$, which is exactly the distribution seen in (3.9).

3.1.2 Atypical subsequences

The main problem we want to solve is finding atypical *subsequences* of long sequences. Consider a sequence $\{x_n, n = -\infty, \ldots, \infty\}$ from a finite alphabet $\mathcal{A}$ (where in this section $\mathcal{A} = \{0, 1\}$. The sequence is generated according to a probability law $\mathcal{P}$, which is known. In this sequence is embedded (infrequent) finite subsequences $\mathcal{X}_i = \{x_n, n = n_i, \ldots, n_i + l_i - 1\}$ from the finite alphabet $\mathcal{A}$, which are generated by an alternative probability law $\tilde{\mathcal{P}}_\theta$. The probability law $\tilde{\mathcal{P}}_\theta$ is unknown, but it might be known to be from a certain class of probability distributions, for example parametrized by the parameter θ. Each subsequence $\mathcal{X}_i$ may be drawn from a different probability law. The problem we consider is to isolate these subsequences, which we call atypical subsequences. In this section, as above, we will assume both $\mathcal{P}$ and $\tilde{P}_\theta$ are binary iid.

The solution is very similar to the one for variable length sequences above. The atypical subsequences are encoded with the universal source coder from [39] with a code length $L_a = lH(\hat{p}) + \frac{1}{2}\log l$. The start of the sequence is encoded with an extra symbol '.' which has a code length $-\log P('.')$ and the length is encoded in $\log^* l$ bits. In conclusion we end up with exactly the same criterion as (3.5), repeated here

$$D(\hat{p}\|p) > \frac{\tau + \frac{3}{2}\log l}{l} \tag{3.16}$$

The only difference is that τ has a slight different meaning.

There is one additional issue here. Since the length of subsequence is not specified, an atypical sequence of length l can be a subsequence of a sequence of length of length $\tilde{l} > l$. The question therefore is how to choose the "correct" length of an atypical sequence, or stated differently, where exactly does an atypical sequence start and end? Also for this problem descriptive length can provide an answer. Namely, the sequence should be divided into typical and atypical segments so as to *minimize the total code length*. We call this *segmentation*. This is similar problem to that considered by Merhav [53], but we will not

dwell further into this problem right now.

For the subsequence problem, a central question is what the probability is that a given sample x_n is part of an (intrinsically) atypical subsequence. Notice that there are infinitely many subsequences that can contain x_n, and each of these have a probability of being atypical given by Theorem 1.

It is difficult to analyze exactly what happens during segmentation. However, we can obtain an upper bound as follows. Let us say that X_n has been determined to be part of an atypical sequence $\mathcal{X}_i$. It is clear that the sequence $\mathcal{X}_i$ must also be atypical according to (3.16). Therefore, we can upper bound the probability $P_A(X_n)$ that X_n is part of an atypical sequence with the probability of the event (3.16), using the approximate criterion (3.6),

$$\exists n_1 \leq n < n_1 + l : \left| \frac{\sum_{i=n_1}^{n_1+l-1} X_i - p}{\sqrt{pql}} \right| > \sqrt{2\tau \ln 2 + 3 \ln l}$$

We can rewrite this as

$$\exists n_1 \leq n < n_1 + l : \left| \frac{\sum_{i=n_1}^{n_1+l-1} X_i - p}{\sqrt{pql}} \right| > \sqrt{2\tau \ln 2 + 3 \ln l}$$

$$\exists n_1 \leq n < n_1 + l : \left| \sum_{i=n_1}^{n_1+l-1} X_i - p \right| > \sqrt{lpq \ln 2(2\tau + 3 \log l)}$$

We could upper bound this with a union bound using Theorem 1. However, it is quickly seen that this does not converge. The problem is that the events in the union bound are highly dependent, so we need a slightly more refined approach. The following result answers Question 1 and partially question 2

Theorem 3. Consider the case $p = \frac{1}{2}$. The probability $P_A(X_n)$ that a given sample X_n is part of an atypical subsequence is upper bounded by

$$P_A(X_n) \leq (K_1 \sqrt{\tau} + K_2) 2^{-\tau} \tag{3.17}$$

for some constants K_1, K_2.

Proof See Appendix A.1. □

There are two important implications of Theorem 3. First is that for τ sufficiently large, $P_A(X_n) < 1$, and in fact $P_A(X_n)$ can be made arbitrarily small for large enough τ. This is an important theoretical validation of Definition 1 and the resulting criterion (3.5), repeated here for reference

$$D(\hat{p}\|p) > \frac{\tau + \frac{3}{2}\log l}{l},$$

and (3.6). If the theory had resulted in $P_A(X_n) = 1$ then everything would be atypical, and atypicality would be meaningless. The fact that this is not trivially satisfied is shown by Proposition 1 just below. What that Proposition says is that if in the above equation instead of $\frac{3}{2}\log l$ we had had $\frac{1}{2}\log l$, then everything would have been atypical. Now, $\frac{1}{2}\log l$ corresponds to "forgetting" that the length of an atypical sequence also needs to be encoded for the resulting sequence to be decodable. Thus, it is the strict adherence to decodability that has lead to a meaningful criterion. So, although decodability at first seems unrelated to abnormality detection, it turns out to be of crucial importance. Similarly, at first the term $\frac{3}{2}\log l$ may have seen arbitrary. However, this is just (within a margin) sufficient to ensure that not everything becomes atypical.

The second important implication of Theorem 3 is that it validates the meaning of τ. The way we introduced τ was as the number of bits needed to encode the fact that an atypical sequence starts, and therefore we should put $\tau = -\log P(\text{atypical sequence starts})$. Theorem 3 confirms that τ has the desired meaning for purely random sequences. And the reasons this is not trivial is that τ was chosen from the probability of an atypical *sequence*, while Theorem 3 gives the probability of a *sample* being atypical.

Proposition 1. Consider the case $p = \frac{1}{2}$. Suppose instead of (3.7) we use the criterion

$$\left| \frac{\sum_{i=1}^{l} X_i - p}{\sqrt{\frac{1}{4}l}} \right| > \sqrt{2\tau \ln 2 + \alpha \ln l} \tag{3.18}$$

(with $\alpha = 3$ giving (3.7)). Then if $\alpha \leq 1$, the probability that a given sample X_n is part of

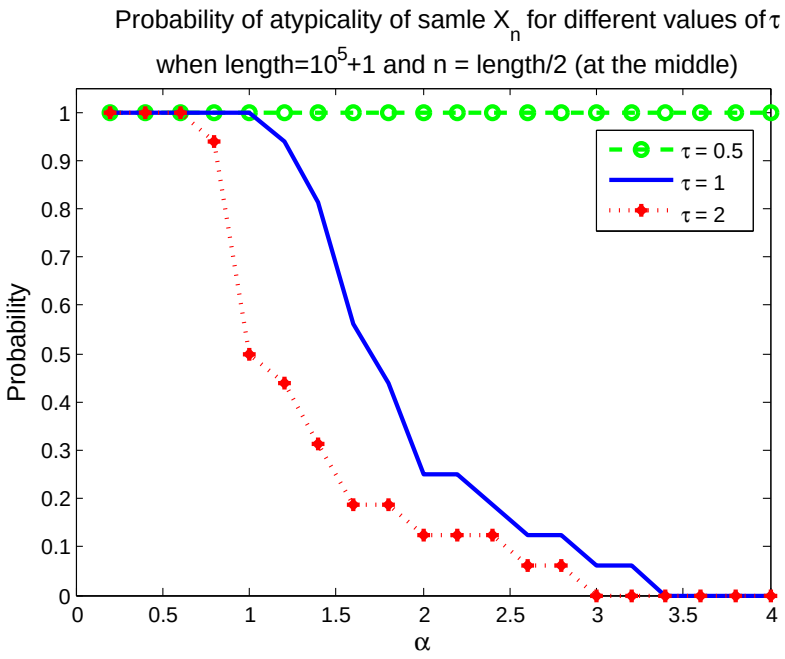

Figure 3.2: Transition between divergence and convergence as a function of α

an atypical subsequence is $P_A(X_n) = 1$.

Proof See Appendix A.2. □

Theorem 3 states that for $\alpha = \frac{3}{2}$ $P_A(X_n) < 1$ (convergence), while Proposition 1 shows that for $\alpha = \frac{1}{2}$ $P_A(X_n) = 1$ (divergence). There is a gap between those values of α that is hard to fill in theoretically. We have therefore tested it out numerically, see Fig. 3.2. Of course, testing convergence numerically is not quite well-posed. Still the figure indicates that the phase transitions between divergence and convergence happens right around $\alpha = 1$.

3.1.3 Recursive coding

Instead of using Definition 1 directly, we could approach the problem as follows. First the sequence is encoded with the typical code. Now, if the distribution of the sequence is in agreement with the typical code, the results should be a sequence of iid binary bits with $P(X_i = 1) = \frac{1}{2}$ [39], i.e., a purely random sequence; and this sequence cannot be further

encoded. We can now try if we can further encode the sequence with a (universal) code. If so, we categorize the sequence as atypical. Let l^* be the length of the sequence after typical coding. In (3.4) the typical and atypical codelengths are therefore

$$\begin{aligned} L_t &= l^* - \log \epsilon \\ L_a &= l^* H(\hat{p}^*) + \frac{3}{2} \log l^* - \log \epsilon + \tau^* \end{aligned} \tag{3.19}$$

Here $\hat{p}^*$ is the estimated p for the *encoded* sequence. Now

$$l^* = l \left(\hat{p} \log \frac{1}{p} + (1 - \hat{p}) \log \frac{1}{1-p} \right) \tag{3.20}$$

$$l^* H(\hat{p}^*) \sim l H(\hat{p}) \tag{3.21}$$

$$\log l^* = \log l + \log \left(\hat{p} \log \frac{1}{p} + (1 - \hat{p}) \log \frac{1}{1-p} \right) \tag{3.22}$$

$$\sim \log l \tag{3.23}$$

The argument for (3.21) is as follows (without doing detailed calculations): if we encode a sequence with a "wrong" code and then later re-encode with the "correct" code (for the induced statistic), the result is the same as originally encoding with the correct code. Thus the criterion (3.4) and (3.19) are *approximately* equivalent. We can state this as follows

Definition 1 can be applied to encoded sequences instead of the original data.

This of course ignores all integer constraints, block boundaries etc. But the importance of this statement is that it is sometimes easier to operate on (partially) encoded sequences simply because the amount of data has already been reduced, and the problem has been standardized: as such, we do not need to know the typical codebook or even the model of typical data since everything under the typical model has been reduced to a stream of iid binary digits, and atypicality algorithms can therefore be applied to data streams without knowledge of what is the original data. It also means that theoretical results such as Theorem 3 where we assume typical data is iid uniform has general applicability.

However, first encoding the sequence and then doing atypicality detection also has disadvantages in a practical, finite length setting. Atypical subsequences become embedded in typical sequences in unpredictable ways. For example, it could be difficult to determine where exactly an atypical sequence starts and ends.

3.2 General Binary Case

Return to the problem considered at the start of Section 3.1 where we are given a sequence x of fixed length l and we need to determine if it is atypical. In the iid case, the solution is given by (3.1). In the non-iid case, we could assume that the atypical model is given by some probability law $\tilde{\mathcal{P}}_\theta$, and then develop a hypothesis test against the typical model given by a probability law $\mathcal{P}$. The issue with this is that if we allow very complex atypical models, we can always find one that fits the data better than the typical model – the well known Occam's razor problem [39]. Rissanen's MDL [43, 44, 54] is a solution to this problem, and Definition 1 is based on this idea.

3.2.1 Finite State Machines

Rissanen [42] defines the complexity of a sequence x^l in the class of FSM (finite state machine) sources by

$$I(x^l) = \min\{-\log P(x^l|f_j) + \log^* j + c\} \tag{3.24}$$

where $f_1, f_2, \ldots$ is a sequence of state machines. Except for integer constraints, this is a valid descriptive length, and can therefore be used in Definition 1. This is a natural extension of the iid case considered in Section 3.1. As opposed to Kolmogorov complexity, this complexity could actually be calculated, although with high complexity. It is mostly useful for theoretical considerations. One result is the following generalization of Theorem 1

Theorem 4. Assume that the typical distribution is iid uniform. If the atypical descriptive length is given by (3.24) with a maximum number of states independent of l, the probability

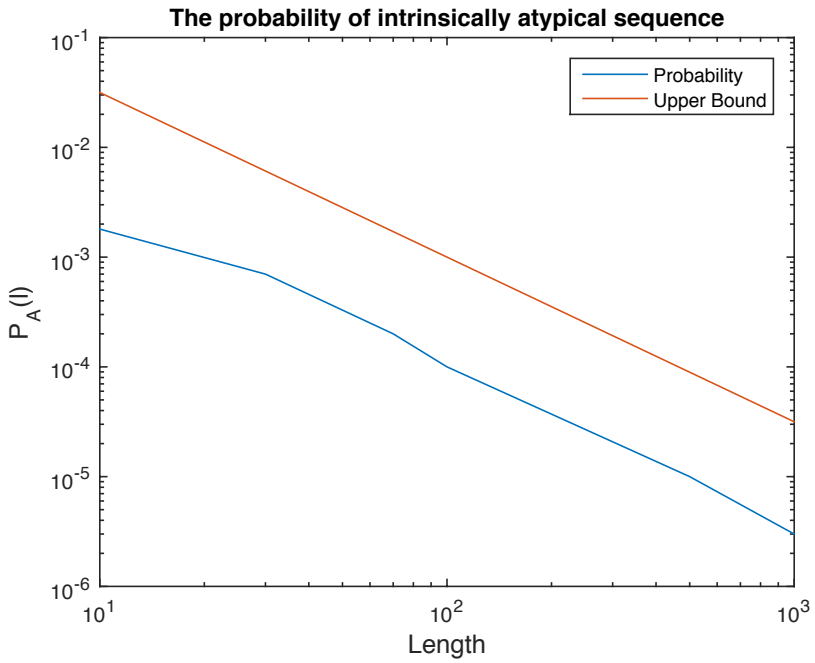

Figure 3.3: Probability of an intrinsically atypical sequence. The typical distribution is iid uniform, and for detection of atypical sequences the CTW algorithm has been used (Section 3.2.2).

of an intrinsically atypical sequence $P_A(l)$ satisfies

$$\lim_{l\to\infty} \frac{\ln P_A(l)}{-\frac{3}{2}\ln l} = 1 \tag{3.25}$$

Proof See Appendix A.3. □

The theorem shows that looking for more complex explanations for data does not essentially increase the probability of intrinsically atypical sequences. Fig. 3.3 (compare with Fig. 3.1) confirms this experimentally. The atypical detection is based on CTW, which as explained in Section 3.2.2, is a good approximation of FSM modeling. On the other hand, if one of the FSM models do in fact fit the data, the chance of detecting the sequence is greatly increased, although hard to quantify. If we think of intrinsically atypical sequences as false alarms, this shows the power of the methodology.

Since FSM sources has the same $P_A(l)$ as in the iid case, it seems reasonable to conjecture

that Theorem 3 is still valid, that is $P_A(X_n) < 1$ for sufficiently large τ, which is clearly an essential theoretical property of atypicality. However, as Theorem 3 does not follow directly from Theorem 1, to verify the conjecture requires a formal proof.

3.2.2 Atypical Encoding

In terms of coding, Definition 1 can be stated in the following form

$$C(x|\mathcal{P}) - C(x) > 0$$

Here $C(x|\mathcal{P})$ is the code length of x encoded with the optimum coder according to the typical law, and $C(x)$ is x encoded 'in itself.' As argued in Section 3.1, we need to put a 'header' in atypical sequences to inform the encoder that an atypical encoder is used. We can therefore write $C(x) = \tau + \tilde{C}(x)$, where τ is the number of bits for the 'header,' and $\tilde{C}(x)$ is the number of bits used for encoding the data itself. For encoding the data itself an obvious solution is to use a universal source coder. There are many approaches to universal source coding: Lempel-Ziv [39, 55, 56], Burrows-Wheeler transform [57], partial predictive mapping (PPM) [58, 59], or T-complexity [60–66], and anyone of them could be applied to the problem considered in the current section. The idea of atypicality is not linked to any particular coding strategy. In fact a coding strategy does not need to be decided. We could try several source coders and choose the the one giving the shortest code length; or they could even be combined as in [67]. However, to control complexity, we choose a single source coder. The most popular and simplest approach to source coding is perhaps Lempel-Ziv [39, 55, 56]. The issue with this is that while it is optimum in the sense that $\limsup_{l\to\infty} \frac{C(x^l)}{l} = H(X)$ wp 1, the convergence is very slow. According to [68] $E\left[\frac{C(x^l)}{l}\right] - H(X) \sim \frac{1}{\log l}$ while $\mathrm{Var}\left[\frac{C(x^l)}{l}\right] \sim \frac{1}{l}$. Thus, Lempel-Ziv is poor for short sequences, which is exactly what we are interested in for atypicality.

Here we will use the Context Tree Weighing (CTW) algorithm [69]. The CTW approach has some advantages in our setup: it is a natural extension of the simple example considered in

Section 3.1, it allows estimation of code length without actually encoding, there is flexibility in how to estimate probabilities. Importantly, it can be seen as a practical implementation of the FSM based descriptive length used in Section 3.2.1.

3.2.3 Typical Encoding and Training

While universal source coding is an obvious choice for atypical encoding, this is not quite as obvious for typical coding. The typical encoding is precisely specified by $C(x|\mathcal{P})$, but if we do not know the typical probability law $\mathcal{P}$ we have to estimate it. We assume, for now, that we are given a single long typical sequence and from this we have to construct an estimate $\hat{\mathcal{P}}$. One possibility is that we have an application specific model $\mathcal{P}_\theta$ that depends on a few parameters θ; in that case this is a standard estimation problem. We will focus on the case when no specific model of $\mathcal{P}$ is given. In principle then $\hat{\mathcal{P}}$ could be given be an estimate of the various joint PMFs. However, this does not give a very feasible method. Rather, we can estimate the conditional probabilities $p(x_n = 1|x_{n-1}, x_{n-2}, \ldots, x_{n-N})$, where $s = x_{n-1}, x_{n-2}, \ldots, x_{n-N}$ is called the context. If the source has finite memory these probabilities characterize the source, and otherwise they could give a good approximate model. The issue is that there are 2^N possible contexts, so for N even moderately large the amount of training data required to get just reasonable estimates for every context is huge. Realistically, therefore not every context can be observed.

Let us write in general the typical encoding of a sequence x as $C(x|y)$, where y is training data and the encoder is a universal source coder. To understand what this means, we have to realize that when x is encoded according to $C(x|\mathcal{P})$, the coding probabilities are *fixed*. Thus, when we estimate the coding probabilities, those estimates should be for the typical data, but not be affected by the atypical data – we need to 'freeze' the source coder. However, because the training data is likely incomplete, the freezing should not be too hard. Furthermore, the encoder $C(x|y)$ is not a universal source coder, but rather a training based fixed source coder, and the implementation can be quite different from a universal source coder.

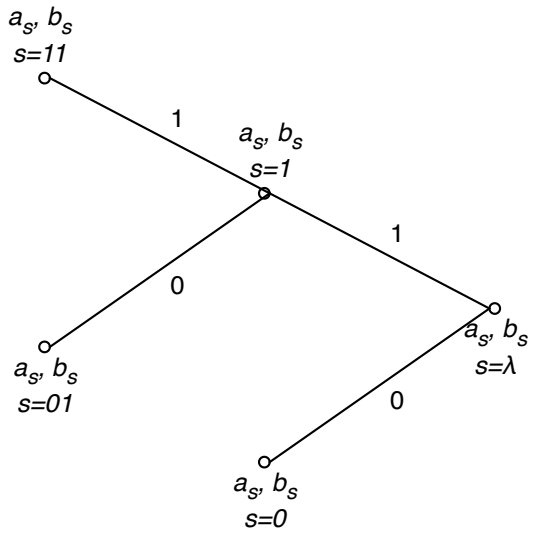

Figure 3.4: Example context tree.

We will suggest one algorithm based on the principle of the CTW algorithm. This naturally complements using the CTW algorithm for atypical encoding, but could also be used with other atypical encoders.

The algorithm is based on estimating $P(x_n = 1|s)$ for contexts s. The estimate for a given s is clearly best done with the KT-estimator [39, 69] $P(x_n = 1|s) = \frac{b_s+1/2}{a_s+b_s+1}$; these are estimated from the training data t, but unaffected by the test sequence x. The complication is that not every context s might be seen and that long contexts s are rarely seen so that the estimates $P(X_n = 1|s)$ might be more accurate for shorter contexts. We solve that with the weighting idea of the CTW algorithm; the weights can be thought of as having a prior distribution on different models. We can summarize this as follows. For every context s, the subsequence associated with s could either be memoryless or it could have memory. We call the former model M_1 and the latter M_2. The CTW algorithm uses a prior distribution, weights, on the models $P(M_1) = P(M_2) = \frac{1}{2}$. The basic idea is to weigh with $P(M_1|t)$ and $P(M_2|t)$ instead of $\frac{1}{2}$.

The algorithm is best described through an example. We have trained the algorithm with t, resulting in the context tree seen in Fig. 3.4. Now suppose we want to find a coding

distribution $P(1|010)$ for the actual data. We begin at the root and calculate

$$\begin{aligned}
P(1|010,t) &= P(1|010,M_1)P(M_1|t) + P(1|010,M_2)P(M_2|t) \\
P(M_i|t) &= \frac{P(t|M_i)P(M_i)}{P(t)} = \frac{P(t|M_i)}{P(t|M_1)+P(t|M_2)} \\
P(t|M_1) &= P_e(a_s,b_s) \\
P(t|M_2) &= P_w^{0s}(t)P_w^{1s}(t) \\
s &= \lambda \quad \text{(empty context)}
\end{aligned}$$

under model M_1 the data is memoryless, so

$$P(1|010,M_1) = P(1|M_1) = \frac{b_s+1/2}{a_s+b_s+1}$$

To find $P(1|010,M_2)$ we look in the 0-node of the context tree. Here we calculate similarly

$$\begin{aligned}
P(1|010,M_2) = P(1|01,t) &= P(1|01,M_1)P(M_1|t) + P(1|01,M_2)P(M_2|t) \\
P(M_i|t) &= \frac{P(t|M_i)P(M_i)}{P(t)} = \frac{P(t|M_i)}{P(t|M_1)+P(t|M_2)} \\
P(t|M_1) &= P_e(a_s,b_s) \\
P(t|M_2) &= P_w^{0s}(t)P_w^{1s}(t) \\
s &= 0
\end{aligned}$$

and again under model M_1 the data is iid, so

$$P(1|01,M_1) = P(1|M_1) = \frac{b_s+1/2}{a_s+b_s+1}$$

and so on. Now from the context tree it is seen that 01 has not been seen before; then the context 010 has not been seen either. Then we have

$$P(1|010,M_2) = \frac{1}{2}.$$

No more look-up is needed, and the calculation has completed.

The algorithm can be implemented as follows. We run the standard CTW algorithm on the training data. To freeze, in each node we can pre-compute $P(M_1|t)$ and $P(1|s, M_1)$. This is the only data that needs to be stored. While the algorithm is described from the root and up, implementation is simpler (no recursion) from the top and down to the root.

Often the source coder might be trained with many separate sequences, rather than one long sequence. This is not an issue, but care has to be taken with the startup for each sequence. The original CTW paper [69] assumes that a context of length at least D is available prior to the start of the sequence. The paper [70] solves this by introducing an indeterminate context. A context may start with an indeterminate context, but at most once. With multiple training sequences this could happen more than once. A better approach is therefore to use the start of each training sequence purely as a context (i.e., not code it). This wastes some training bits, but if the sequences are long the loss is minor. A different case would be if we need to find short atypical sequences rather than subsequences. In that case a more careful treatment of start of sequences would be needed.

We would also like to implement the typical coding in parallel processing. For the typical encoding after training, this is straightforward, as each sequence can be encoded independently and therefore processed independently. For training, parallel processing can be done by processing each training sequence independently; if there is only one training sequence, this can be split into multiple sequences, so that the end of a sequence is used as the initial context for the next sequence, which is equivalent to processing the whole sequence sequentially. At the end, the different estimates need to be combined. However, the probabilities cannot be directly combined. Rather, the counts a_s and b_s are added up for the different

parts, and the $P_e(a_s, b_s)$ are calculated for the total counts by

$$\begin{aligned} P_e(a_s, b_s) &= \frac{\frac{1}{2} \times \cdots \times (a - \frac{1}{2}) \times \frac{1}{2} \times \cdots \times (b - \frac{1}{2})}{1 \times 2 \times \cdots \times (a + b)} \\ &= \frac{(2(a_s + 1))!(2(b_s + 1))! 4^{-a_s - b_s - 2}}{(a_s + 1)!(b_s + 1)!(a_s + b_s + 2)!} \\ &= \frac{\Gamma\left(a_s + \frac{3}{2}\right) \Gamma\left(b_s + \frac{3}{2}\right)}{\pi \Gamma(a_s + b_s + 3)} \end{aligned}$$

rather than sequentially. From these estimates, the weighted estimates are as usually calculated by

$$P_w^s = \frac{1}{2} P_e(a_s, b_s) + \frac{1}{2} P_w^{0s} P_w^{1s}$$

Thus, each parallel process just counts zeros and ones in the context tree, while the calculations of the probabilities must be done centrally. It is therefore not clear that parallel implementation is that efficient.

Other universal source coders can probably also be frozen. For example, in dictionary based algorithms, the dictionary could be frozen. Freezing the encoder is essential in implementing atypicality. A simulation confirming this is shown in Fig. 3.5. The CTW algorithm is trained with a three-state non-IID Markov chain with transition probability $[.05.950; 0.05.95; .950.05]$. This Markov chain mostly generates the following pattern: [1 0 1]. The test sequence is generated by another three-state non-IID Markov chain with the same transition probability [.05 .95 0 ; 0 .05 .95 ; .95 0 .05], but this Markov chain mostly generates the following pattern: [1 0 0]. With the non-frozen algorithm the code length difference between typical and atypical encoding is so small that it can easily be missed, although the difference in the patterns themselves in the raw data is clearly visible to the naked eye. The reason the non-frozen algorithm does not work is that it quickly learns the new [1 0 0] pattern. Any good source coder would do that including LZ. This is advantageous to source coding, but in this case it means missing a very obvious atypical pattern.

If a very large amount of data is used for training, the complexity can become very high, mainly in terms of memory. Namely, all contexts might be observed, and the context tree

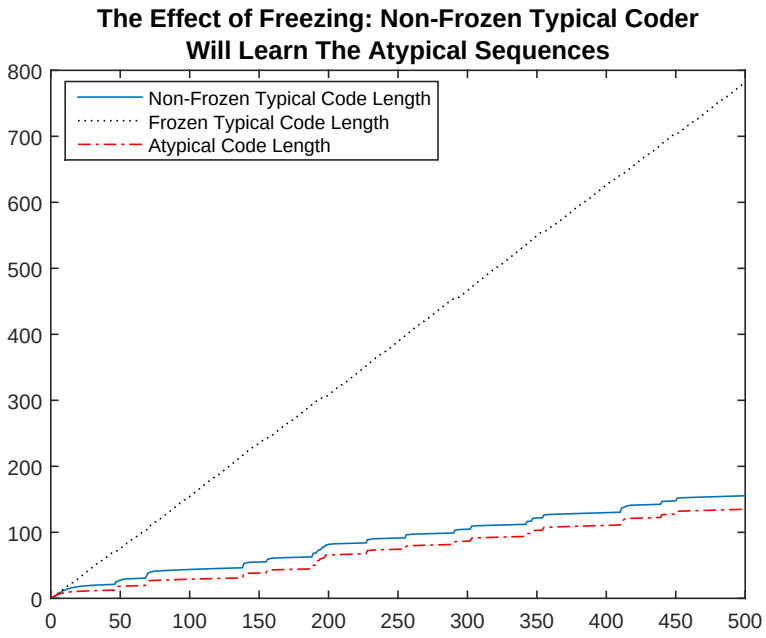

Figure 3.5: The importance of freezing source coding when testing for atypicality.

completely filled out. For example, suppose the typical data is actually iid. That means that every string $x_1, \ldots, x_N$ is seen with equal probability. For the CTW algorithm that means that every node of the context tree will be filled out, and the number of nodes with depth D is 2^{D+1}. For dictionary based algorithms, it means that the dictionary size becomes huge. What is needed is some algorithm that not only estimates the unknown parameters, but also the model (e.g., iid). One way could be to trim the context tree (or dictionary), but we have not looked into this in detail.

3.2.4 Atypical subsequences

For finding atypical subsequences of a a long sequences, the same basic setup as in the previous sections can be used. Let $\mathcal{X}(n, l) = (x_n, x_{n+1}, \ldots, x_{n+l-1})$ be a subsequence of $\{x_n, n = 0, \ldots, \infty\}$ that we want to test for atypicality. As in Section 3.1.2 the start of a sequence needs to be encoded as well as the length. Additionally the code length is minimized

over the maximum depth D of the context tree. The atypical code length is then given by

$$L_A(\mathcal{X}(n,l)) = \min_D \left(-\log P_w^\lambda(D) + \log^* D \right) + \log^* l + \tau$$

Here $P_w^\lambda(D)$ denotes the probability at the root of the context tree [69] of depth D. Since we are also interested in finding short sequences, how the encoding is initialized is of importance, and for atypical coding we therefore use the algorithm in Section II of [70].

For typical coding we use either a known fixed model and Shannon codes, or the algorithm in Section 3.2.3 when the model is not known; when we encode $\mathcal{X}(n,l)$ we use $x_{n-D}, \ldots x_{n-1}$ as context for x_n (we can assume $n > D$). Equivalently, we can encode the total sequence $\{x_n, n = 0, \ldots, \infty\}$ (with the algorithm from Section 3.2.3); let $L(n)$ be the codelength for the sequence $x_0, \ldots, x_n$. Then we can put $L_T(\mathcal{X}(n,l)) = L(n+l-1) - L(n)$.

We need to test every subsequence of every length, that is, we need to test subsequences $\mathcal{X}(n,l)$ for every value of n and l. For atypical coding this means that a new CTW algorithm needs to be started at every sample time. So, if the maximum sequence length is L, L separate CTW trees need to maintained at any time. These are completely independent, so they can be run on parallel processors.

The result is that for every bit of the data we calculate

$$\Delta L(n) = \min_l L_A(\mathcal{X}(n,l)) - L_T(\mathcal{X}(n,l)) \tag{3.26}$$

This implementation clearly is very complex.

3.3 Optimality Theorem

We will next discuss how well atypicality can detect anomalies. An FSM can have several classes, that is, sub-FSM, where once the FSM has entered the class, it will not escape again. There are of course only a finite number of such classes. We will say that two FSM

are *distinct* if they have no identical classes. Suppose that typical data is generated by some FSM, and anomalous sequence is generated by another FSM. If they have an identical class, there is a non-zero probability that both FSM will be in that class. The sequences generated will be statistically identical, and there is no way to determine which FSM generated the sequence: the anomaly will not be detected no matter what method is used. In terms of anomaly detection, that would be called a miss. However, in terms of atypicality, this is correct: since the sequence could have been generated by the typical model, it should be classified as typical. This emphasizes the point we made in the introduction: atypicality is a property of data, while anomaly is related to how the data was generated. This is also why we use intrinsically versus extrinsically atypical, rather than false alarm and miss, to characterize performance. However, if the two FSM are distinct, atypicality used for anomaly detection satisfies the following:

Theorem 5. Let the atypical codelength be given by (3.24). Suppose that the typical model is an FSM. Let the atypicality detector be given an anomalous sequence generated by an FSM distinct from the typical FSM. Then as the length of the sequence $l \to \infty$, the probability of detecting the anomaly converges to 1, while the probability of false alarm converges to 0.

Proof In order to use Theorem 4, rather than directly coding with typical and atypical coders, we will use an equivalent approach. First the sequence is encoded with the typical coder. If the sequence is indeed typical, the result will be an iid, uniform sequence [39]. We now compare the identity coder with an atypical coder, which adheres closely to the starting point of Kolmogorov-Martin-Löf randomness.

If the sequence is typical Theorem 4 proves that the probability that it is detected as atypical converges to zero.

If the sequence is generated by an FSM distinct from the typical FSM, the result of using the typical coder on the sequence results in another sequence that can be seen as generated by a combined FSM. How this combined FSM looks like and how many states it has has a

complicated relationship with the original FSMs; however, we just need that it is an FSM The original FSM will eventually reach a stationary distribution [71], which is the stationary distribution of a class, and the combined FSM therefore will also reach a steady state. Furthermore, the sequence will be ergodic for the steady state solution (but not necessarily for the total FSM). Let the entropy rate of of the steady state solution (i.e., chosen class) of the original FSM be $\mathcal{H}_o$; this is not necessarily the same as the entropy rate of the complete random process. The average codelength per bit after using the typical coder now is $L_t = \mathcal{H}_o + \epsilon$, where we know [39] that $\epsilon > 0$ strictly – here we use that the two FSM are distinct. On the other hand, if the atypical codelength (3.24) is used, the average codelength according to Theorem [42, Theorem 2]

$$L_a \leq \mathcal{H}_o + \frac{k^*}{2l} \log l + O\left(\frac{1}{l}\right) \tag{3.27}$$

where k^* is the number of states in the combined FSM. The bits needed to tell the receiver that the sequence is encoded with the atypical coder is included in the $O\left(\frac{1}{l}\right)$ term. For sufficiently large l, $\frac{k^*}{2l} \log l + O\left(\frac{1}{l}\right) < \epsilon$. While this is a statement about average codelength, the actual codelength for a specific sequence will converge to the average in probability due to ergodicity. Therefore for any δ, for sufficiently large l, we can ensure that $P_D > 1 - \delta$ and $P_{FA} < \delta$ (probability of detection and false alarm, respectively), which is the claim of the theorem. □

Theorem 5 can be interpreted so that for the problem of anomaly detection in the class of FSM, atypicality is asymptotically optimal: it will with probability 1 classify sequences correctly into anomalous and non-anomalous. The assumption of FSM is certainly restrictive, in particular the assumption that also anomalies are generated by an FSM: we would clearly want to also detect anomalies not from FSM, for example non-stationary sequences. But still, having the result in the general class of FSMs is a stronger statement than any other method for anomaly detection that we know of. We also believe that Theorem 5 is valid in a much wider class of sequences, due to the generality of Definition 1, but a formal proof is

difficult.

3.4 Algorithms

In this section, two major methodologies that play an important rule in atypicality search are introduced. First a fast detection algorithm is outlined, and then segmentation process is formally stated.

3.4.1 Fast Detection

The issue with the the approach in Section 3.1.2 is complexity, as it requires us to check *every* possible subsequence. Suppose we are interested in subsequences with length $1 \leq l \leq l_{\max}$. Then for every sample time n we need to calculate the codelengths for all sequences starting at n with length $1 \leq l \leq l_{\max}$. Fortunately, source coders are usually recursive, so the codelengths for sequences of length $l < l_{\max}$ are automatically calculated on the way to calculate the codelength of the sequence of length $l_{\max}$. Thus, assuming a linear complexity of the source coder, the complexity is $O(l_{\max})$ for every sample. This can still be significant if $l_{\max}$ is large.

One idea to speed up processing is to divide the total sequence into non-overlapping subsequences of length $l_{\max}$. Each is then further divided into two non-overlapping subsequences of length $\frac{l_{\max}}{2}$, and then again into non-overlapping subsequences of length $\frac{l_{\max}}{4}$ and so on. It is now easily seen that the complexity per sample is reduced to $O(\log l_{\max})$, a considerable saving if $l_{\max}$ is large. Assume $l_{\max}$ is a power of two, call the subsequences used for this approach *detection subsequences*, and let their lengths be $l_1, l_2, \ldots, l_m$, where $l_1 = l_{\max}, l_2 = l_1/2, \ldots, l_m = 1$.

The key is that we would like to have comparable performance to the exact algorithm where every subsequence is examined. Suppose we have a (single) atypical subsequence x of length

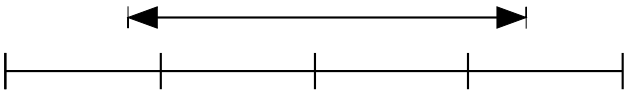

Figure 3.6: Fast Detection. For any atypical subsequence (with the arrows) there exists at least one fully embedded detection sequence of a quarter length.

$l \leq l_{\max}$. It is easy to see that we can always find some l_k, so that x is fully contained in a subsequence of length l_k, and there is at least one subsequence of length $l_k/4$ fully contained in x, which is illustrated in Fig. 3.6.

Let us first discuss the issues when we use the simple IID coder in Section 3.1. The subsequence x could be found from either one of the detection subsequences partially overlapping with x, or from one of the subsequences of length $l_t = l_k/4$ fully contained in x. We will focus on optimizing the latter case. If $\hat{p}$ is far from p, it is not an issue detecting x from a shorter subsequence. The interesting case is when $\hat{p}$ is close to p. We notice, most clearly from (3.6), that the threshold for $\hat{p}$ depends on the length l. If $\hat{p}$ is just above the threshold, it has a low probability of being detected from the shorter subsequence. From Fig. 3.6 it can be seen that $l < 4l_k/4$. Therefore, to compensate for the shorter length, we can repeat the detection subsequence of length $l_k/4$ four times, and encode this repeated sequence. *This is the central idea in our fast algorithm.* Now, the codelength is quite accurately given by $L = l_t H(\hat{p}) + \frac{1}{2}\log l_t$. Thus, if we repeat the sequence r times, the codelength is $\hat{L} = r l_t H(\hat{p}) + \frac{1}{2}\log r l_t$ quite accurately, or

$$\hat{L} = rL - \frac{(r-1)}{2}\log l_t + \frac{1}{2}\log r \tag{3.28}$$

Thus, after we have calculated the codelength for the detection subsequence of length l_t, we calculate the codelength for the repeated sequence from (3.28), which is much faster than actually repeating. For the typical codelength, the length with repetition is simply $\hat{L}_t = rL_t$.

If we ignore the randomness of $\hat{p}$, the above approach with $r = 4$ will find 100% of actual atypical sequences. When $\hat{p}$ is actually estimated, the detection probability is less than one. The detection probability depends on many parameters, but one example can be seen in

Fig. 3.7, generated as follows. A long typical sequence with $p = 0.3$ is generated. Inside the typical sequence, an atypical segment of length $l = 100$ with random permutation of $l\hat{p}$ 1's is randomly located where $p_0 < \hat{p} < 1$ and p_0 is the the solution to the atypicality criterion 3.5 which for $\tau = 5$ gives $p_0 = 0.52$. Therefore this setup ensures that exact processing is able to detect the inserted atypical subsequence with certainty. Then fast algorithm is used to find the atypical segment, and miss probability is calculated using Monte Carlo method. In spite of running a large number of simulations, not a single miss with $r = 4$ was found. On the other hand, when there is no repetition, $r = 1$ the miss probability is large, confirming that repetition is needed.

The issue with using $r = 4$ is that the false alarm probability is also high. It is important to realize the role played by false alarms. After the fast algorithm have localized candidate atypical subsequences, they are analyzed with the exact algorithm calculating (3.26). False alarms therefore do not show up in the final result, but they affect computational complexity. This can be measured through the probability $P_A(X_n)$ of Theorem 3: this is the probability that a given sample is part of some atypical sequence; from Theorem 3 we know that this probability decreases exponentially with τ. Suppose for example that $P_A(X_n) = 10^{-3}$ for the fast algorithm. That means that about 1 out of 1000 samples are found to be atypical. These needs to be analyzed with the exact algorithm. But this is still 1000 times faster than analyzing every sample. Fig. 3.8 shows this probability for different repetition factor r. To understand this figure, suppose that we want the final intrinsic $P_A(X_n)$ to be 10^{-6} (we can think of this as the ultimate false alarm rate). We then choose $\tau = 15$. Then $P_A(X_n) = 10^{-1}$ for $r = 4$ and $P_A(X_n) = 10^{-3}$ for $r = 2$. Thus, $r = 4$ only reduces the number of computations for exact processing with a factor 10; and considering the extra overhead in transitioning from fast processing to exact processing, there might not be any saving. It can be seen that for $r = 4$ computational complexity is not reduced much, whereas $r = 2$ reduces computations by a large factor, while still having a high detection probability, Fig. 3.7. Therefore, $r = 2$ might be a good tradeoff between detection probability and complexity.

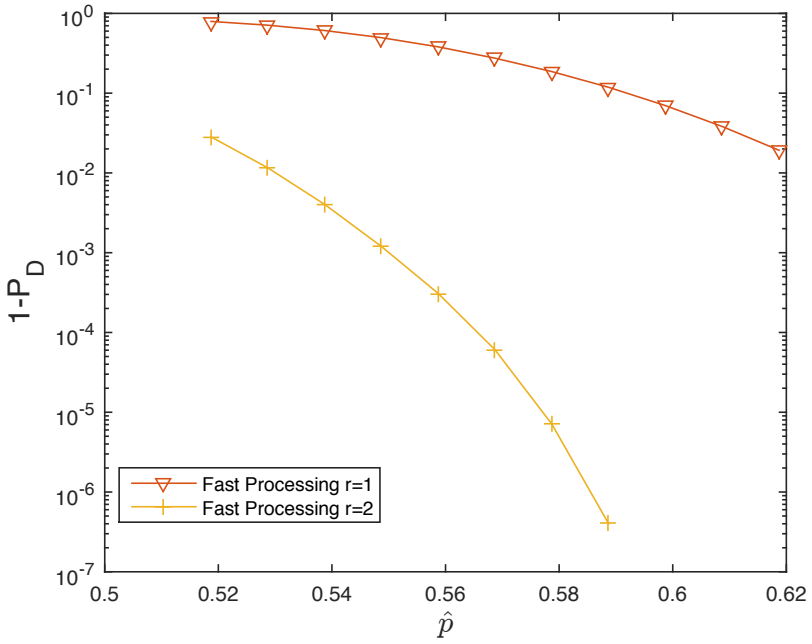

Figure 3.7: Miss probability $1 - P_D(x^l)$ for $\tau = 5$, $p = 0.3$ and $l = 100$. The probability for $r = 4$ is so small that our simulation didn't find a single case out of 10^7 runs, so the curve cannot be shown.

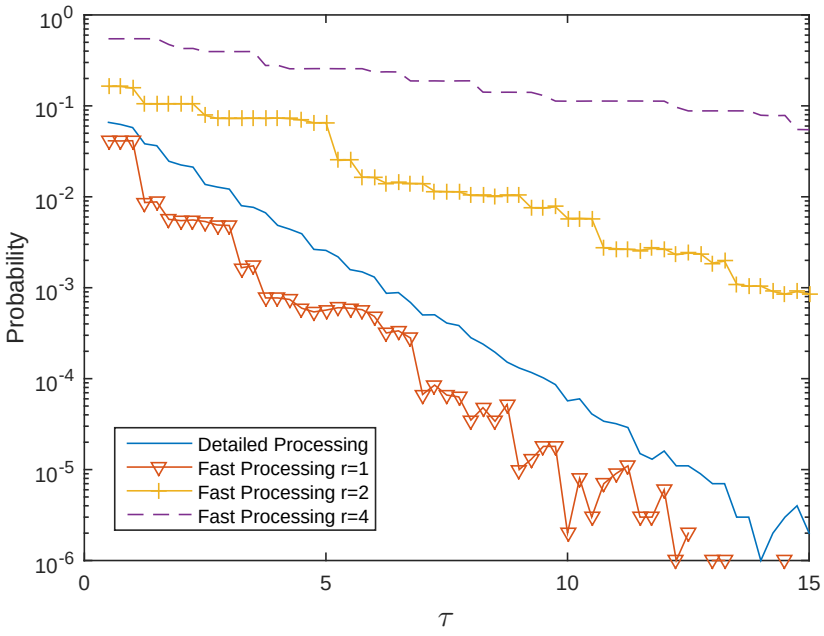

Figure 3.8: Intrinsic atypicality probability $P_A(X_n)$.

We would like to generalize this methodology beyond IID sequences, specifically using the CTW algorithm. The simplest way, as above, is to simply repeat every detection sequence r times. However, this creates two issues. The first is that doing the coding on the repeated sequence increases complexity r times. The second is that now the exact same sequence is repeated . On the other hand, if we had an actual sequence four times longer, it would not exactly repeat four times, rather just be statistically similar. When a good source coder sees the exact sequence repeated, it should code it efficiently, and the CTW indeed does. Thus, the codelength for the repeated sequence in general will be much shorter than for a sequence r times longer.

We will first consider the complexity issue. Recall that the CTW algorithm is based on estimating $P(x_n = 1|s)$ for contexts s. The estimate for a given s is done with the KT-estimator [39, 69] $P(x_n = 1|s) = P_e(a_s, b_s) = \frac{b_s+1/2}{a_s+b_s+1}$, where b_s and a_s are the number of 1s respectively 0s seen with context s. In every node, the algorithm calculates

$$P_w^s = \frac{1}{2}P_e(a_s, b_s) + \frac{1}{2}P_w^{0s}P_w^{1s}$$

Rather than explicitly repeating, we modify $P_e(a_s, b_s)$ as in (3.28) by

$$\log \hat{P}_e = \log P_e(a_s, b_s) + \frac{(r-1)}{2}\log(a_s + b_s) - \frac{1}{2}\log r \tag{3.29}$$

and then we calculate $\hat{P}_w^s = \frac{1}{2}\hat{P}_e + \frac{1}{2}\hat{P}_w^{0s}\hat{P}_w^{1s}$. It turns out this is not exactly the same as repeating the sequence, and that it is less optimistic than simply repeating the sequence. But it still considerably underestimates the codelength. To understand why in a more analytical way, assume the input sequence is IID uniform and that it is coded with a CTW with $D = 2$. Thus, the sequence is coded with the IID coder, and then split into l_0 samples with context 0 and l_1 samples with context 1, $l_1 + l_0 = l$, which are each coded with an IID coder. The

codelength (weighted probability) in the root then is within a good approximation

$$\begin{aligned} P_w^\lambda &\approx \frac{1}{2}2^{-lH(\hat{p})}\frac{1}{\sqrt{l}} + \frac{1}{2}2^{-l_0H(\hat{p}_0)}\frac{1}{\sqrt{l_0}}2^{-l_1H(\hat{p}_1)}\frac{1}{\sqrt{l_1}} \\ &= \frac{1}{2}P_a\frac{1}{\sqrt{l}} + \frac{1}{2}P_b\frac{1}{\sqrt{l_0l_1}} \end{aligned} \tag{3.30}$$

If we now replace the estimated probabilities with (3.29), we instead get

$$\begin{aligned} P_w^\lambda &\approx \frac{1}{2}2^{-rlH(\hat{p})}\frac{1}{\sqrt{rl}} + \frac{1}{2}2^{-rl_0H(\hat{p}_0)}\frac{1}{\sqrt{rl_0}}2^{-rl_1H(\hat{p}_1)}\frac{1}{\sqrt{rl_1}} \\ &= \frac{1}{2}P_a^r\frac{1}{\sqrt{rl}} + \frac{1}{2}P_b^r\frac{1}{\sqrt{2rl_0l_1}} \end{aligned} \tag{3.31}$$

Here is then what happens. Usually $P_b > P_a$, as the more complex model fits data better. In (3.30) the first term is usually still larger, as the "complexity factor" $\frac{1}{\sqrt{l_0l_1}} \approx \frac{2}{l}$ will reduce it sufficiently. Now in (3.31) the lifting to the r-thj power is quite dramatic (P_a and P_b are very small), and we could then have $P_b^r \gg P_a^r$, so that the second term will dominate. It will look as though the non-IID is better than the IID coder.

We are compensating for this in a simple way. If, after coding the detection sequence without repetition, the IID coder is better, then the non-IID coder will be be used for the repeats. Explicitly

- If $P_e(a_s, b_s) > P_w^{0s}P_w^{1s}$, put
$$\hat{P}_w^s = \hat{P}_e$$
- Otherwise
$$\hat{P}_w^s = \frac{1}{2}\hat{P}_e + \frac{1}{2}\hat{P}_w^{0s}\hat{P}_w^{1s}$$

This is done in every node of the context tree. Figure 3.9 shows an example of performance. An IID sequence with $p = \frac{1}{2}$ of length $l = 100$ is coded with a CTW coder with $D = 10$ and repeated twice; this is compared with the codelength of a sequence with $l = 200$. If we just plainly repeat the sequence, we obtain a very optimistic codelength, but with the proposed

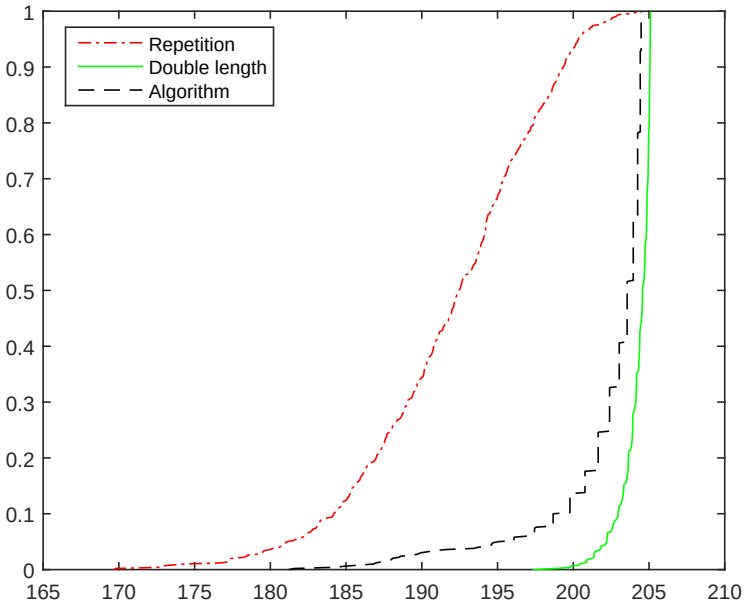

Figure 3.9: CDF of codelength.

algorithm we obtain a more realistic codelength.

We will analyze complexity in a little more detail. Let the computational cost of calculating the CTW be lK_C, where K_C is some constant that depends on $D_{\max}$. Assume $l_{\max}$ is a power of two, and that further are not interested in sequences below a certain minimum sequence length $l_{\min}$, also a power of two. For a direct computation, the complexity per sample is $l_{\max}K_C$; the minimum sequence length is not useful. Now for the fast algorithm, sequences are found from detection sequences down to a quarter length. Therefore, the longest sequences that need to be calculated is $l_{\max}/4$. Similarly, the minimum detection sequence length is $l_{\min}/4$. As for direct detection, not every sequence need to be compressed separately. For example, if the CTW of length l is calculated, on the way to calculating this, also the CTW for sequences of length $l/2, l/4, \ldots$, with the same starting point are calculated. A closer inspection shows that only half the sequences need to calculated explicitly. The accounting therefore is as follows. The total sequence of length N is divided into $\frac{N}{l_{\max}}$ segments. In each segment we calculate 4 CTWs of length $\frac{l_{\max}}{4}$, 4 CTWs of length $\frac{l_{\max}}{8}$, 8 CTWs of length $\frac{l_{\max}}{16}$

and so on, so

$$C = K_C \frac{1}{N} \cdot \frac{N}{l_{\max}} \left(l_{\max} + \frac{l_{\max}}{2} (\log l_{\max} - \log l_{\min} - 1) \right)$$
$$= \frac{K_C}{2} (\log l_{\max} - \log l_{\min} + 1)$$

For example, if $l_{\max} = 2^{20}$ and $l_{\min} = 2^7$, complexity is reduced by a factor around 150,000 by the fast algorithm! However, as mentioned, the output of the fast processing has to be next analyzed by the exact algorithm, and this mainly depends on $P_A(X_n)$. Potentially, this is more important for total computation complexity.

3.4.2 Segmentation

After finding initial atypical subsequences using general or fast detection, the next step is to find the exact start and end point of detected atypical subsequences, i.e., segmentation. For instance, if $\Delta L(n) < -\tau$ with $\Delta L(n)$ given by (3.26), this indicates that there is an atypical subsequence that includes x_n. However, it does not indicate exactly which subsequence, i.e., where it starts and ends. Therefore, after detection of atypical subsequences, the task still remains of dividing samples into typical and atypical.

3.4.2.1 IID case

Suppose we detect an atypical sequence of length l. The further complication here is that the atypical sequence can be a subsequence of a sequence of length $\tilde{l} > l$. The question therefore is how to choose the "correct" length of an atypical sequence, or stated differently, where exactly does an atypical sequence start and end? Also for this problem, the descriptive length can provide an answer. Namely, the sequence should be divided into typical and atypical segments so as to *minimize the total code length.*

Assume the detected atypical sequence starts at n_1 and ends $n_2 = n_1 + l - 1$, so for all

$n \in [n_1, n_2]$ the sample x_n is part of an atypical sequence of at least length l. As stated above, segmentation is done by minimizing the total code length. To see how to do this we can write the code length (3.4) as

$$\sum_{i=n_1}^{n_2} x_i \log \frac{1}{\hat{p}} + \sum_{i=n_1}^{n_2} (1 - x_i) \log \frac{1}{1 - \hat{p}} + \frac{3}{2} \log(n_2 - n_1 + 1) + \tau \tag{3.32}$$

On the other hand, if we encode the same sequence with the typical code, the code length is

$$\sum_{i=n_1}^{n_2} x_i \log \frac{1}{p} + \sum_{i=n_1}^{n_2} (1 - x_i) \log \frac{1}{1 - p}$$

For sake of argument, suppose $\hat{p} > p$. Consider increasing n_2 by 1. Now, if $n_2 - n_1$ is reasonably large, this will affect $\hat{p}$ and the log term only slightly. We therefore ignore these changes, which is equivalent to assume $\hat{p}$ is known in advance. With these assumptions, if $x_{n_2+1} = 1$, the total code length is decreased by the amount

$$\Delta = \log \frac{1}{\hat{p}} - \log \frac{1}{p} = \log \frac{p}{\hat{p}} < 0$$

and if $x_{n_2+1} = 0$ the total code length is increased by the amount

$$\Delta = \log \frac{1}{1 - \hat{p}} - \log \frac{1}{1 - p} = \log \frac{1 - p}{1 - \hat{p}} > 0$$

Thus as we increase n_2 the total code length will alternately increase and decrease, and the total code length will look like an (asymmetric) random walk. Therefore, there are infinitely many local minimum for the code length, and it will take infinitely long to tell which is the global minimum. Random walk theory could be used to find a reasonable threshold, but as this is not the actual problem we are trying to solve we will not proceed with this exact analysis. Returning to the real problem, with the log-term and the estimated $\hat{p}$, the same procedure can be used, and in fact with more sharp decision points. Namely, suppose n_2 is increased and past the "actual" end of the atypical sequence. Then not only will Δ

Algorithm 1 Binary IID atypicality segmentation

- The algorithm should take the data (with length N), $P(X=1)$ of typical part, τ, initial $n_1^{(1)}$ and $n_2^{(1)}$as an input.
- The following optimization parts can be done in iteration ($k = 2 : MaxIteration$): For $n_1^{(k)}$ less than $n_1^{(k-1)}$, minimize the following total code length:

$$\begin{aligned}
&\sum_{i=1}^{n_1^{(k)}-1} x_i \log\frac{1}{p} + \sum_{i=1}^{n_1^{(k)}-1} (1-x_i)\log\frac{1}{1-p} \\
&+ \sum_{i=n_1^{(k)}}^{n_2^{(k-1)}} x_i \log\frac{1}{\hat{p}} + \sum_{i=n_1^{(k)}}^{n_2^{(k-1)}} (1-x_i)\log\frac{1}{1-\hat{p}} \\
&+\frac{3}{2}\log(n_2^{(k-1)} - n_1^{(k)} + 1)
\end{aligned}$$

now use new n_1 (i.e. $n_1^{(k)}$) and for $n_2^{(k)}$ greater than $n_2^{(k-1)}$, minimize the following total code length

$$\begin{aligned}
&\sum_{i=n_1^{(k)}}^{n_2^{(k)}} x_i \log\frac{1}{\hat{p}} + \sum_{i=n_1^{(k)}}^{n_2^{(k)}} (1-x_i)\log\frac{1}{1-\hat{p}} \\
&+\frac{3}{2}\log(n_2^{(k)} - n_1^{(k)} + 1) \\
&+ \sum_{i=n_2^{(k)}+1}^{N} x_i \log\frac{1}{p} + \sum_{i=n_2^{(k)}+1}^{N} (1-x_i)\log\frac{1}{1-p}
\end{aligned}$$

generally be positive, but the log-term will also increase, and $\hat{p}$ will start to deviate from the "actual" value, which means that the other bits will be encoded less efficiently. On the other hand, finding n_1 and n_2 is no longer independent. They are coupled both through the log-term and through $\hat{p}$, and the coupling through $\hat{p}$ is complicated. What we propose is to alternately find n_1 and n_2, i.e., optimize n_1, then optimize n_2 for the given n_1, then optimize n_1 for the given n_2 and so forth. The optimization can be done as for the case when $\hat{p}$ is known. In algorithm 1, a way of atypicality segmentation implementation for binary IID case is proposed.

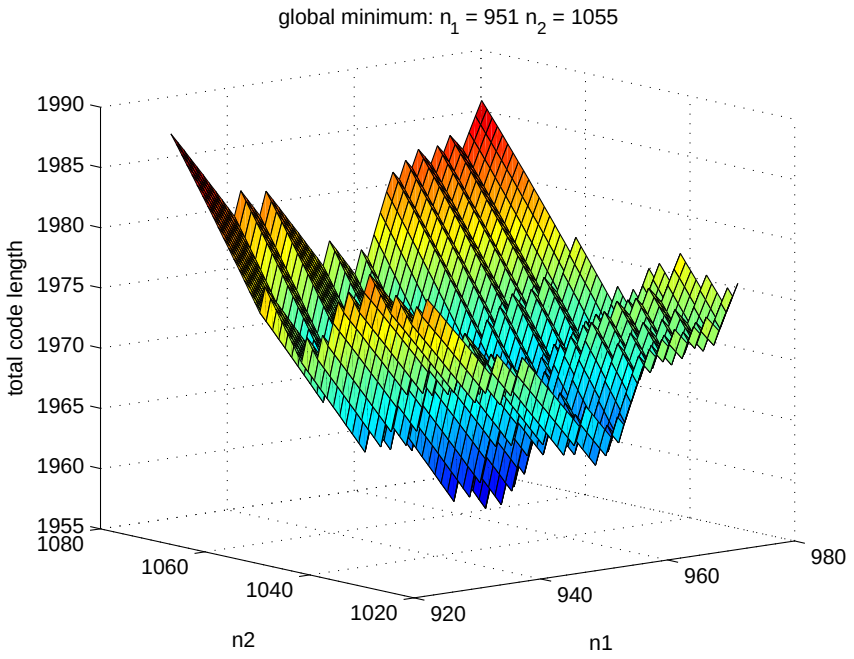

Figure 3.10: Total code length versus $n_1$and n_2.

In order to check the performance of algorithm 1, we applied on a simulated example. In this example, a 2000-sample binary IID sequence is generated according to $P(X = 1) = 0.5$ and the interval $[950, 1050]$ is replaced with another binary IID sequence generated according to $P(X = 1) = 0.9$. In Fig. 3.10 we can see the total code length versus different value of n_1 and n_2. For this example, n_1 and n_2 according to the algorithm 1 were 951 and 1055, which is exactly the point that total code length has its global minimum.

As we discussed before, we assume that the detected length of an atypical sequence in the detection phase is smaller that the real length, i.e. $n_1^{(1)} > n_1$ and $n_2^{(1)} < n_2$. So we need some statistic to check how often this assumption is valid. Therefore we decided to run this experiment repeatedly to calculate the relative CDF of difference between real start/end point of atypical sequence and segmentation output (i.e., relative CDF of $n_1 - n_1^{(k)}$ and $n_2 - n_2^{(k)}$). In Fig. 3.11 we can see these CDFs. According to this figure, as we expected, relative CDF of n_1 is left-sided and also relative CDF of n_2 is almost right-sided. A closer inspection of the data and algorithm shows in the cases that $n_1 - n_1^{(k)} > 0$ or $n_2 - n_2^{(k)} < 0$,

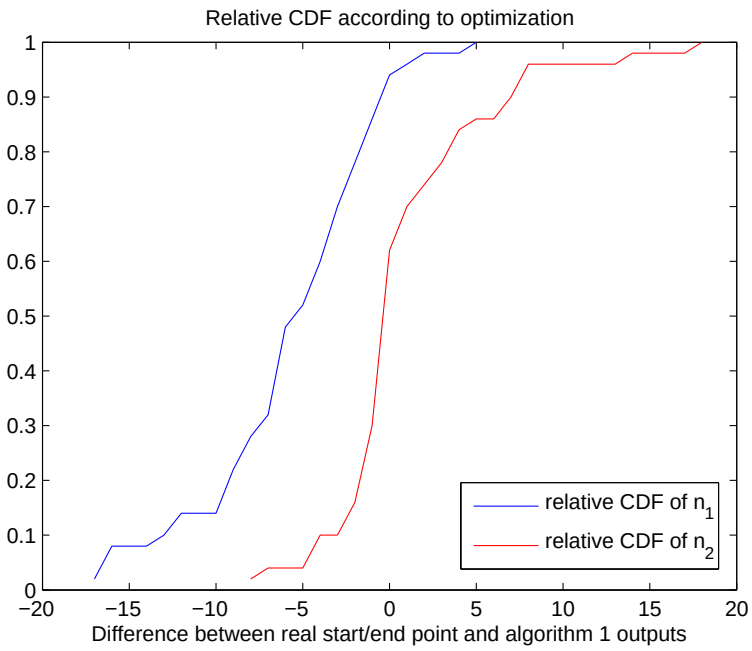

Figure 3.11: Relative CDF of $n_1$and n_2

we had $n_1 - n_1^{(1)} < 0$ or $n_2 - n_2^{(1)} > 0$, but the global minimum of total code length had happened some place where the final length of the detected atypical sequence was greater than real length of atypical sequence. As we can see, even in these cases, the difference between real start/end point and algorithm 1 outputs are less than 8 bits.

3.4.2.2 Non–IID case

The method is straightforwardly generalized to the non-IID case, summarized in Algorithm 2.

Simulated Example

To check the result, we simulate a non-IID binary example here. We simulate data with 451 samples. Samples from 1 to 200 and 252 to 451 are generated according to random variable generator with $p = \frac{1}{3}$, and samples from 201 to 451 are repetitions of "1 0 0". The initial

Algorithm 2 Binary non-IID atypicality segmentation

- The algorithm should take the data (with length N) and τ as an input.
- For each n, find the smallest l satisfying $\Delta L(n,l) < -\tau$ in (3.26). This phase is detection, so we will have initial $n_1^{(1)}$ and $n_2^{(1)}$.
- The following optimization parts can be done in iteration ($k = 2 : MaxIteration$): For $n_1^{(k)}$ less than $n_1^{(k-1)}$, minimize the following total code length:

$$\begin{aligned} CL_1 = \{&\log(P_\omega^\lambda(n_1^{(k)})) - \log(P_\omega^\lambda(1)) \\ &+ \min_D[-\log P_\omega^\lambda(D, n_1^{(k)} + 1 : n_2^{(k-1)})] + \log^*(n_2^{(k-1)} - n_1^{(k)})\} \end{aligned}$$

now use new n_1 (i.e. $n_1^{(k)}$) and for $n_2^{(k)}$ greater than $n_2^{(k-1)}$, minimize the following total code length

$$\begin{aligned} CL_2 = \{&\log(P_\omega^\lambda(N)) - \log(P_\omega^\lambda(n_2^{(k)} + 1)) \\ &+ \min_D[-\log P(D, n_1^{(k)} + 1 : n_2^{(k)})] + \log^*(n_2^{(k)} - n_1^{(k)})\} \end{aligned}$$

estimation for atypical segment will be $n_1 = 222$ and $n_2 = 250$. We then use Algorithm 2. In Fig. 3.12 $\Delta L(n)$, CL_1 and CL_2 are shown, so by minimizing total code length, we end up with $n_1 = 198$ and $n_2 = 251$ which is close to the actual separation points.

3.5 Unsupervised Case

In the original definition of atypicality, Definition 1, the model of the typical model was assumed known. This was extended to a training based case in Section 3.2.3, where instead of having a known model, the algorithm is given some data that is *known* to be typical, that is, contains no atypical samples. The training based case is useful in many contexts. For example, in the medical case we might use data from people we know does not have a given decease. However, this does not always work. For example, in the medical case, people might have some decease but be symptom free, so that they are falsely classified as 'typical.' Then the atypicalities will be classified as typical.

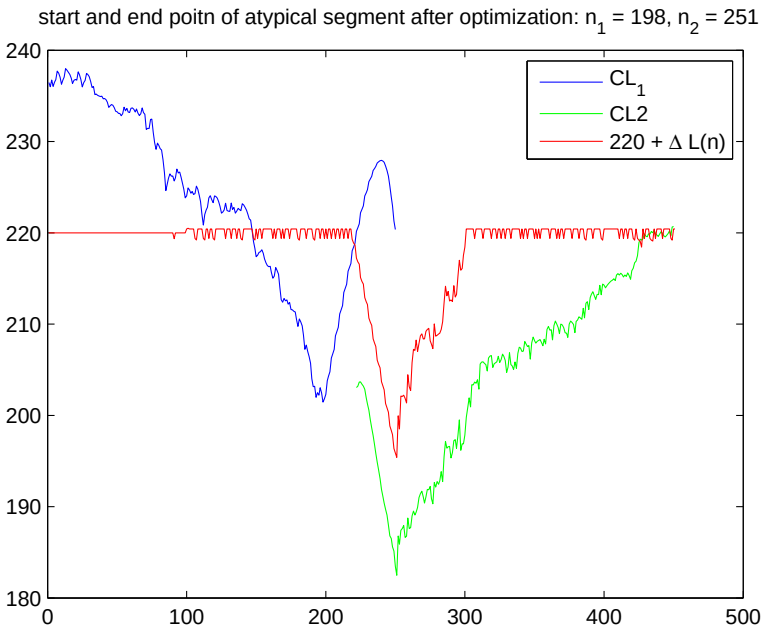

Figure 3.12: $\Delta L(n)$, CL_1 and CL_2 which are used to find n_1 and n_2.

We therefore consider the unsupervised case. We are given a collection of data, of which some might be atypical. We then need to separate it into typical and atypical data. It is immediately clear that without constraints, this is not a well-posed problem. The partition can be arbitrary. The most important constraint in the atypicality problem is that atypical data must be rare. But rare is not the same as atypical. Rare is essentially the same as unlikely, that is the same as surprise. We therefore need further constraints, and in the following we will discuss some possible approaches based on the information theory fundamentals.

3.5.1 Definitions

Suppose we have an (ordered) set $\mathcal{S}$ of sequences x_i. For every sequence x_i that we examine, there are three choices (as in [72]): 1) it could belong to our current typical model, 2) it could be atypical, 3) it could extend our typical model. By the latter we mean for example that it could be part of a new cluster of data. We want to distinguish between these cases

in the context of Definition 1. One idea is to use total codelength.

First we need the following definitions. Let $\mathcal{T}$ be a set of sequences, and let C be some source coder algorithm. We say that C is trained on $\mathcal{T}$ if we allow C to adapt to $\mathcal{T}$ and then freeze it. As discussed in Section 3.2.3, freezing is important for atypicality.

Let $\mathcal{A}$ be a non-empty subset of sequences with $|\mathcal{A}| < \frac{1}{2}|\mathcal{S}|$. Let C be a source coder trained on all of $\mathcal{S}$, and let C_t be a source coder trained on $\mathcal{A}^c$. We now use the following definition

Definition 2. The partition $(\mathcal{A}, \mathcal{A}^c)$ is an *atypicality partition* if

1. For every sequence $x \in \mathcal{A}$, $C_t(x) > C_a(x)$.
2. The partition reduces total codelength. That is,
   $\sum_{x\in\mathcal{S}} C(x) > \sum_{x\in\mathcal{A}} C_a(x) + \sum_{x\in\mathcal{A}^c} C_t(x)$

□

The first condition is directly Definition 1. The second condition is specific to the unsupervised case. This is a criterion to distinguish between case 2 and 3 above, based on total codelength.

A simple example best illustrates the idea of Definition 2. Suppose we have a collection of m binary iid sequences of length n. Of these $m-1$ comes from a distribution with $P(X=1) = p_0$ while one comes from a distribution with $P(X=1) = p_1$. Let us assume that p_1 is far from p_0 and that n is reasonably large, so that the p_1 sequence is quite distinct from the rest. It is clear that simply using a single iid source coder for all the sequences results in a long code length for the p_1 sequence. However, an alternative that is often used in machine learning is *clustering*. So, as typical model we could instead use an iid model with two clusters, one centered around p_0 (that is, an estimate of it) and one centered around p_1. To encode sequences in the second cluster, we use the universal source coder from Section 3.1. This requires $nH(\hat{p}_1) + \frac{1}{2}\log n$ bits. To encode it as an atypical sequence also requires $nH(\hat{p}_1) + \frac{1}{2}\log n$ bits. The difference is in the number of bits required to tell the decoder

which decoder to use. For the atypical encoder this is done by inserting the 'header' of τ bits before every atypical sequence. For the clustered model, instead we first transmit m bits indicating the cluster number for each sequence; this sequence itself is encoded with a universal source coder requiring $mH\left(\frac{1}{m}\right) + \frac{1}{2}\log m$ bits. To summarize

$$\begin{aligned}
\sum_{x\in\mathcal{S}} C(x) &= n(m-1)H\left(\hat{p}_0\right) + \frac{1}{2}\log((m-1)n) \\
&\quad + nH\left(\hat{p}_1\right) + \frac{1}{2}\log n + mH\left(\frac{1}{m}\right) + \frac{1}{2}\log m \\
\sum_{x\in\mathcal{A}^c} C_t(x) &= n(m-1)H\left(\hat{p}_0\right) + \frac{1}{2}\log((m-1)n) \\
\sum_{x\in\mathcal{A}} C_a(x) &= nH\left(\hat{p}_1\right) + \frac{1}{2}\log n + \tau
\end{aligned}$$

Thus, the p_1 sequence is atypical if

$$mH\left(\frac{1}{m}\right) + \frac{1}{2}\log m > \tau.$$

Recall that in principle $\tau = -\log P('\text{atypical}')$, and in this case the best estimate of $P('\text{atypical}')$ is m^{-1}. Thus τ may be of the order $\log m$, but since the first term on the left hand side is large, a single p_1 sequence is unlikely to be categorized as typical.

If instead there are k sequences from the p_1 distribution, the calculus changes. These sequences can now be encoded as one long sequences requiring $nkH\left(\hat{p}_1\right) + \frac{1}{2}\log kn$. Encoding them as atypical sequences requires about $nkH\left(\hat{p}_1\right) + \frac{1}{2}k\log n$. As k increases, encoding as a separate cluster quickly becomes advantageous because of the second term. In a little more detail, when encoding as separate atypical sequences, each sequence has its own estimate of $\hat{p}_1$, which reduces codelength slightly. Therefore the k sequences will be atypical if

$$\begin{aligned}
mH\left(\frac{k}{m}\right) + \frac{1}{2}\log m + nkH\left(\hat{p}_1\right) + \frac{1}{2}\log kn \\
> k\tau + \sum_{j=1}^{k} nH\left(\hat{p}_{1,j}\right) + \frac{1}{2}k\log n
\end{aligned}$$

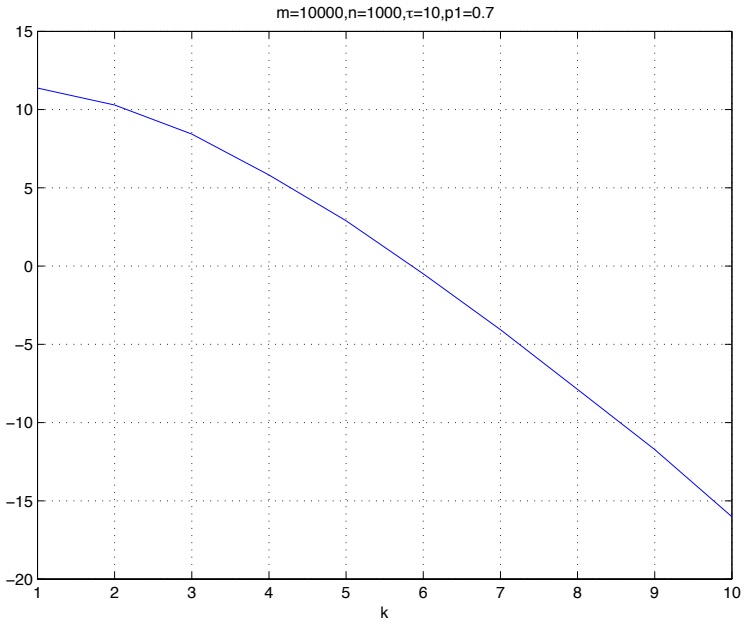

Figure 3.13: Test statistic.

It is hard to predict analytically when this is satisfied. We did one numerical experiment with $m = 10,000, n = 1,000, \tau = 10, p_1 = 0.7$, and for $k > 6$ total codelength is smaller by clustering than by declaring sequences atypical.

Fig. 3.13 shows a numerical plot for some particular parameters. It shows that when there are more than $k = 6$ 'different' sequences out of 10,000, they will be classified as typical rather than atypical. This seems reasonable as $\tau = 10$ implies we expect around $2^{-10} \times 10,000 = 10$ atypical sequences. This just validates that the criterion is in reasonable agreement with common sense.

Now, it is unlikely that an atypicality partition is unique. We make the following reasonable definition of optimality.

- A set $\mathcal{S}$ is said to be *completely typical* if it has no atypicality partitions.
- A *complete atypicality partition* is an atypicality partition $(\mathcal{A}, \mathcal{A}^c)$, where $\mathcal{A}^c$ is completely typical.

- A complete atypicality partition is said to be *minimum* if there exists no partitions with $|\mathcal{A}|$ smaller.
- The *optimum partition* is the one among the minimum partitions that have the smallest codelength per sample in $\mathcal{A}^c$.

3.5.2 Algorithms

Finding an optimum or even a complete atypicality partition is very complex. If there are m sequences, there are 2^m possible partitions. To even check that a set is completely typical most likely requires checking all those partitions. Thus, we resort to heuristic algorithms. The key here is the assumption that atypical data is (very) rare. Thus, the problem is similar to outlier detection. Unfortunately, it seems the problem is sufficiently different that we cannot use standard methodology from outlier detection directly. However, outlier detection could be used as starting point.

To develop algorithms, it is necessary to understand the importance of completeness. To clarify, consider an extension of the simple example above. There m total sequences, of which $m-k-1$ sequences come from a distribution $P(X=1)=p_0$, $k \ll m$ sequences from $P(X=1)=p_1$ and one sequence from $P(X=1)=p_2$; p_0, p_1, p_2 are assumed to be distinct and n large, so that it is fairly clear which sequence comes from which distribution. Let us say that the typical model is simply iid (with no clusters). Then with high likelihood[1] there are four valid atypicality partitions: $\mathcal{P}_0 = \{\text{no atypical sequences}\}$, $\mathcal{P}_1 = \{\text{the } p_2\text{sequence atypical}\}$, $\mathcal{P}_2 = \{\text{the } p_1\text{sequence atypical}\}$, and $\mathcal{P}_3 = \{\text{both the } p_1\text{and } p_2\text{sequences atypical}\}$[2]. The partition $\mathcal{P}_3$ seems like the intuitively correct solution. And indeed, with high likelihood this is the only partition that is complete. For $\mathcal{P}_1$, if we split the remaining sequences into the typical p_0 sequences and atypical p_1 sequences with high probability this will reduce total codelength. Thus, for $\mathcal{P}_1$, $\mathcal{A}^c$ is not complete.

[1]Since the sequences are random, we cannot state anything with certainty. The statements we make can be interpreted the way that they are valid for sufficiently large n by the law of large numbers.

[2]The partition with p_0 sequences atypical is not valid by the condition $|\mathcal{A}| < \frac{1}{2}|\mathcal{S}|$

If we now use a more complex typical model that allows clusters, we have the same four possible atypical partitions. Now both $\mathcal{P}_1$ and $\mathcal{P}_3$ have the potential of being complete partitions. Partition $\mathcal{P}_1$ is complete if it gives the shortest codelength to encode the p_1 and p_0 sequences as a clustered model, rather than splitting p_1 out as atypical sequences; whether or not this happens depends on k and other parameters. If it happens that both $\mathcal{P}_1$ and $\mathcal{P}_3$ are complete, we should choose $\mathcal{P}_1$ since it is minimum.

Our suggested algorithm is as follows. The sequences are coded sequentially. To avoid any bias based on position, the sequences can first be randomly permuted. Because atypical sequences are rare, it is highly unlikely that the first few sequences are atypical (to handle this situation, the algorithm can be run a few times with different permutations, and some sort of combined solution can be found). Thus, from the first few sequences, the typical coder will learn how typical sequences look like. The coding now continues until condition 1 in Definition 2 is satisfied. The first such sequence is declared atypical, preliminarily, and not added to the training set for the typical coder. The algorithm now continues to sort sequences into typical and atypical sequences until $|\mathcal{A}|$ is large. Let $\mathcal{T}$ be the subset of sequences declared typical until that point. If $|\mathcal{A}|$ is large, the idea is that perhaps not all sequences in $|\mathcal{A}|$ are atypical, since atypical sequences are rare. We therefore reexamine $\mathcal{A}$.

The first step is to sort the sequences in $\mathcal{A}$ according to how likely they are to be truly atypical. We suggest the following measure. If a sequence $x \in \mathcal{A}$ is atypical, the idea is that knowledge of any other sequence should not help (much) in coding x. To test this let C_x be a source coder trained on $\mathcal{T} \cup \mathcal{A} - x$. We now sort the sequences in $\mathcal{A}$ in increasing order with respect to $C_x(x) - C_a(x)$. Let $(x_1, \ldots . x_{|\mathcal{A}|})$ be the set of ordered sequences.

The next step is to find the minimum "complete" partition among the partitions $\mathcal{T} \cup \{x_1, \ldots, x_i\}, \{x_{i+1}, \ldots, x_{|\mathcal{A}|}\}$. True completeness is hard to check, but we can at least make sure that the obvious partitions of $\mathcal{T} \cup \{x_1, \ldots, x_i\}$ where we make some of of the x_j atypical are not valid. To this end make the sequence of source coders $C_0, \ldots, C_{|\mathcal{A}|}$, where C_i is a source coder trained on $\mathcal{T} \cup \{x_1, \ldots, x_i\}$. We now run the following procedure

1. Let $k = |\mathcal{A}| + 1$.

2. [Atypicality] For j from k to $|\mathcal{A}|$ check that $C_{k-1}(x_j) > C_a(x_j)$.

3. [Completeness] For j from 0 to $k-1$ check that[3]

$$\exists m : j < m < k : C_j(x_m) < C_a(x_m) \quad \text{OR}$$
$$\sum_{x \in \mathcal{T} \cup \{x_1, \ldots, x_{k-1}\}} C_{k-1}(x) < \sum_{x \in \{x_{j+1}, x_{k-1}\}} C_a(x) + \sum_{x \in \mathcal{T} \cup \{x_1, \ldots, x_j\}} C_j(x)$$

4. If both the conditions in 2 and 3 are satisfied or $k = 1$, STOP. Otherwise, decrease k by 1, and go to 2.

We now define a new atypicality set $\mathcal{A} := \{x_k, \ldots, x_{|\mathcal{A}|}\}$, and go on looking for atypical sequences. We notice that the above procedure never questions the typical set; the typical set can be made larger in the reexamination step, but never smaller. If one or more atypical sequences are among the first coded, they could be included in the typical set. The chance of this happening is low, by the assumption of rareness of atypical sequences. Yet, it may still happen. One way to deal with this is run the procedure a few times with different random permutations of the sequences. If the runs give different typical sets, we take the intersection of the sets to be the set of sequences were are quite certain are typical; let's call this the base typical set. We now make one final runs, where we start by examining sequences in the base typical set.

3.6 Experimental Results

In order to verify the performance of our algorithm, we used three different experiments. In the first we evaluated randomness of sources, in accordance with our starting point of

[3]Notice that for many source coders, for $x \in \mathcal{T}$, $C_{k-1}(x) = C_j(x)$, so those terms cancel out.

Kolmogorov-Martin-Löf randomness. In the second, we looked for infection in human DNA, and in the third we looked for arrhythmia in ECG.

A word about presentation of the results. For the outcome of our method we plot $\Delta L(n)$ given by (3.26). At the same time, we would like to illustrate the raw data. The source in all cases is a stream of bits $x[n] \in \{0, 1\}$. We convert this to $y[n] \in \{-1, 1\}$ (i.e, $y[n] = (-1)^{x[n]-1}$), and then plot $S[N] = \sum_{n=1}^{N} y[n]$; we call this the random walk representation. In our experience, this allows one to quickly assess if there is any obvious pattern in data. If the data is random, the results will look like a typical random walk: both small fluctuations and large fluctuations.

All experimental data and software used is available at http://itdata.hostmadsen.com.

3.6.1 Coin Tosses

In this experiment the typical data is iid binary random. As source of typical data we used experimental coin tosses from [73]. This data consists of 40,000 tosses by two Berkeley undergraduates of a fair coin and the result has 20,217 heads ($X_i = 1$), so $\Pr\{X_i = 1\} = 0.505425 \approx \frac{1}{2}$. Therefore we can consider it as a real binary IID experiment, indeed it is an example of pure random data. In our experiments with this data, we examine the randomness of other types of data.

One type of data one might *think* is purely random are word length changes in a text. In the first experiment, we generate binary data using consecutive word length comparison of part of M. B. Synge's "On the shores of the great sea" in the following manner: If the next word is longer than the current word, 1 is assigned to the binary data, otherwise 0. In the case of two consecutive words with same length, a random 0 or 1 is generated (with a *good* random number generator). We then insert this data in the coin toss data. Since we assume coin tosses data is IID, there is no need to train the CTW and the the code length of the IID case (3.2) can be used for typical coding. Fig. 3.14 illustrates the result of the algorithm

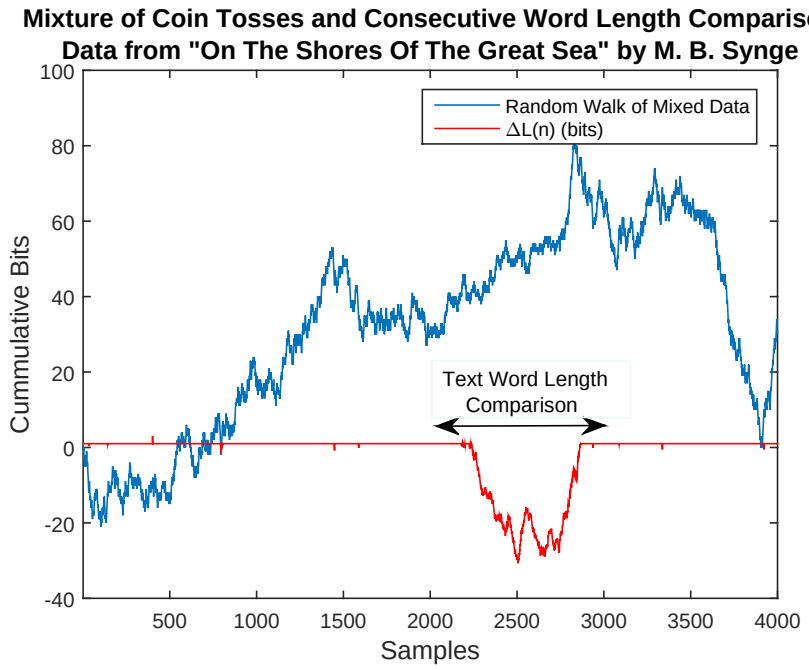

Figure 3.14: Random walk of mixed coin tosses and consecutive word comparison.

on the mixed data. Thus, word length changes are not iid random. Perhaps this is because word length is bounded from above and below, so that there are limits to how long runs of 0s or 1s are possible.

In second experiment, we generated random data with the infamous RANDU random number generator. This was a random number generator that was widely used until it was discovered that it has some clear deviation from randomness. RANDU generates random numbers in the interval $[0, 2^{31} - 1]$, so each number needs 31 bits for binary representation. But instead of using 31 bits for each number, we sum up all the 31 bits and compare it with 15.5 to generate either 0 or 1. Then this data is inserted in the part of coin tosses data. Fig. 3.15 shows that the most atypical segment is where we have inserted data from RANDU random number generator.

In the third experiment, we generate binary data using consecutive heart rate comparison of part of normal sinus rhythm downloaded from MIT- BIH database [74] in the same way as for the text. Fig. 3.15 represents the result of the algorithm on the mixed data. As

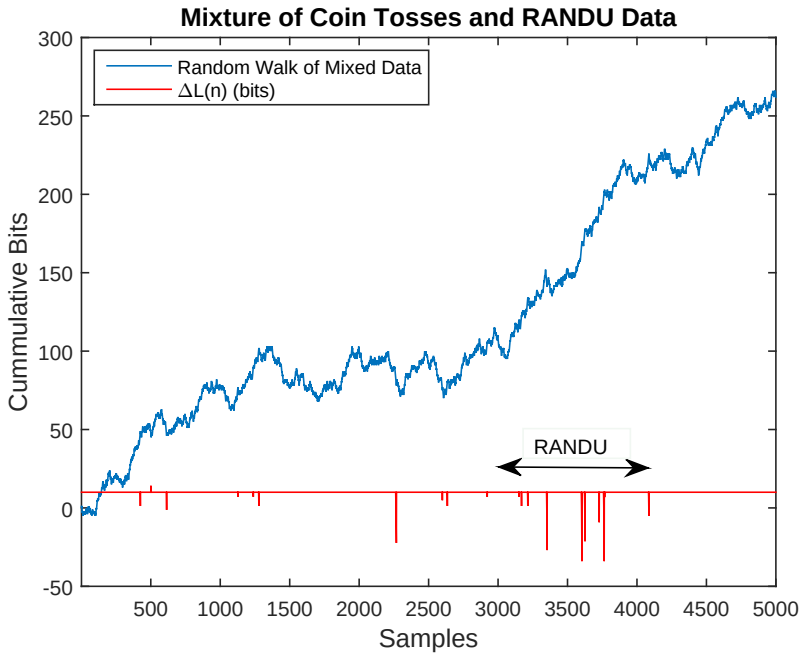

Figure 3.15: Random walk of mixed coin tosses and RANDU.

can be seen the atypicality measure shows a huge difference between iid randomness and randomness of consecutive heart beats. We don't know why this is the case.

3.6.2 DNA

In this collection of experiments, we detect viral and bacterial insertion into human genomic DNA. DNA from foreign species can be inserted into the human genome either through natural processes [75], typically though viral infections, bacterial infections, or through genetic engineering [76]. The inverse also occurs in genetic engineering experiments during the creation of "transgenic" organisms, with the insertion of human DNA into bacteria, yeast, worms, or mice. In the experiments we show here, we have focused on the former case. We train the CTW algorithm on pure human genomic DNA.

The data that we have used was comprised of ˜20 kilobases of human genomic DNA (each sequence from a different chromosome) with either bacterial or viral random DNA sequences

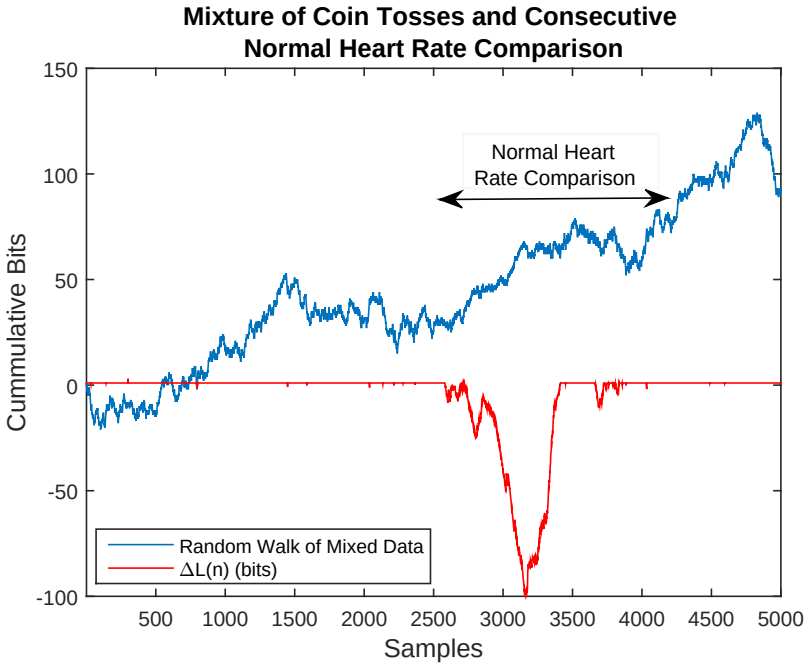

Figure 3.16: Random walk of mixed coin tosses and HRV.

(˜2 kilobases per insertion) inserted. Since our software is too slow to find atypical sequences of length more than a few hundreds, we removed the middle of the insertions. Notice that this actually makes detection harder. We used some of the human DNA for training, but not the same as the test sequences.

In the first experiment we tried to detect short sequences from Streptococcus Pneumoniae (a bacterial infection with a high fatality rate, and a frequent cause of death in the elderly) randomly inserted into larger segments of human genomic DNA. Fig. 3.17 illustrates the result of the experiment. Based on the figure, the inserted Streptococcus Pneumoniae DNA fragment was detected by our algorithms.

In the second experiment we tried to detect HIV inserted into human genomic DNA to mimic viral infection, which is a more realistic experiment since viruses typically insert their DNA into the host genome every time a human obtains a viral infection. Fig. 3.18 illustrates the result of the experiment. As can be seen, the infected viral fragment was detected by our algorithms.

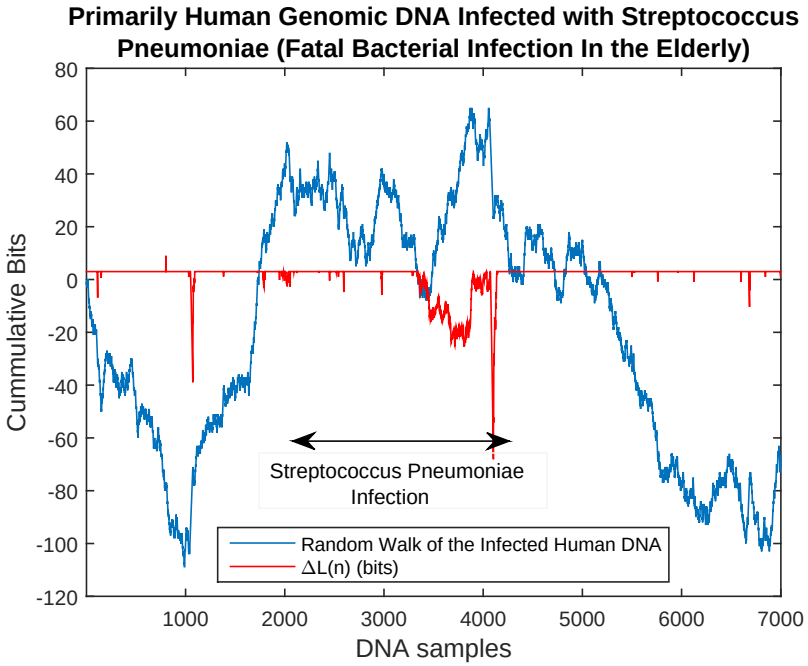

Figure 3.17: Random walk of human DNA with bactrial infection.

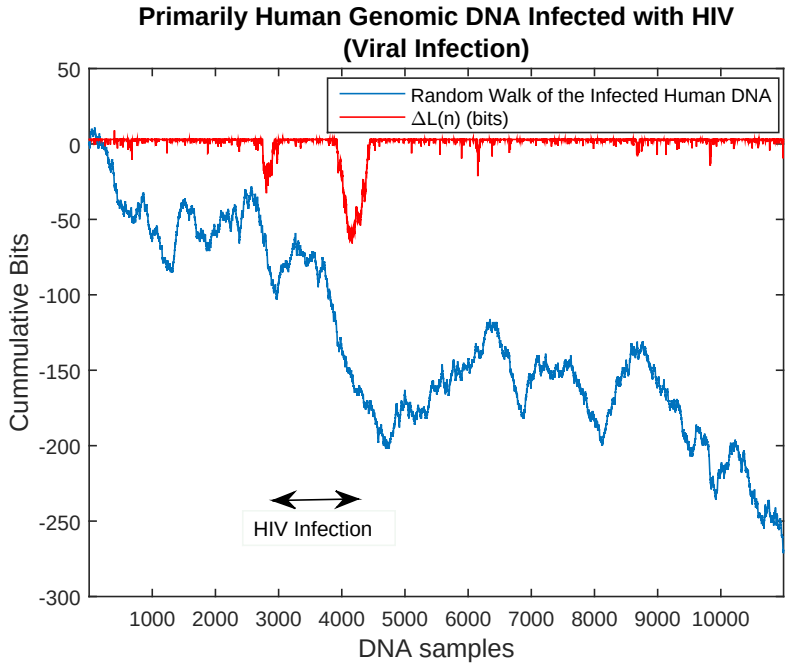

Figure 3.18: Random walk of human DNA with viral infection.

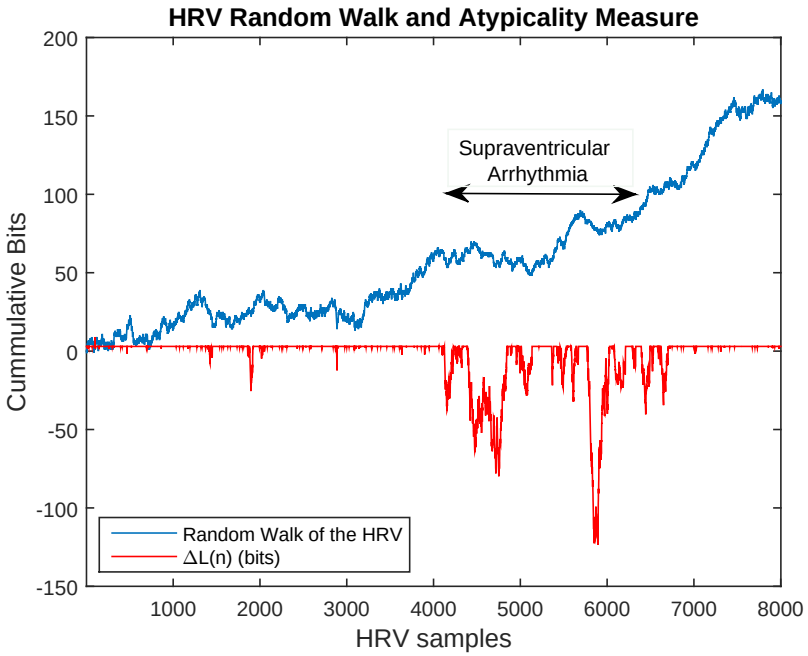

Figure 3.19: Random walk of HRV.

3.6.3 HRV

While HRV (Heart Rate Variability) can be a powerful indicator for arrhythmia [34], the common issue is that it is not known exactly what to look for in the data. Our aim for this application is to use atypicality to localize signs of subtle and complex arrhythmia. In [77] based on our modest goal of localizing a simple type of known arrhythmia, we managed to find premature beats using HRV signal, but here we attempted to detect more subtle arrhythmia. The HRV signals that we used were downloaded from MIT- BIH database [74]. We used "MIT-BIH normal sinus rhythm database (nsrdb)" and "MIT-BIH supraventricular arrhythmia database (svdb)". Encoding of HRV signals were done by same manner as the text word length comparison of subsection 3.6.1. In this experiment, CTW was trained with HRV of normal sinus rhythm, then applied to a HRV signal that has supraventricular rhythms. Fig. 3.19 shows the result of the simulation. The algorithm was able to localize the segment that suffers from abnormal rhythms.

4

Real-Valued Case

We would like to extend definition 1 to real-valued models. The definition is based on *exact* description of data, and lossless source coding rather than lossy (rate-distortion) therefore is the appropriate generalization. Lossless coding of real-valued data is used in for example lossless audio coding [78], so it is not an unusual approach.

Direct encoding of the reals represented as binary numbers, such as done in lossless audio coding, makes the methods too dependent on data representation instead of the underlying data. Instead we will use a more abstract models of (finite-precision) reals. We will assume a fixed point representation with a (large) finite number, r, bits after the period, and an *unlimited* number of bits prior to the period. Assume that the actual data is distributed

according to a pdf $f(x)$. Then the number of bits required to represent x is given by

$$\begin{aligned} L(x) &= -\log \int_x^{x+2^{-r}} f(t)dt \\ &\approx -\log(f(x)2^{-r}) \\ &= -\log(f(x)) + r \end{aligned} \tag{4.1}$$

The approximation is good when $f(x)$ is nearly constant over an interval of length 2^{-r}, and it can be made arbitrarily good with $r \to \infty$. Since we are only interested in comparing different codelengths, say under distributions $f_1(x)$ and $f_2(x)$, the dependency on r cancels out.

One advantage of considering real-valued data rather than binary data is that the wealth of signal processing methods can be applied. This will be the focus of this section. In most signal processing models, the key element is the signal, i.e., the relationship between samples, rather than the exact distribution of individual samples. Often, the randomness is assumed to be Gaussian. In accordance with this, most of our methods will focus on Gaussian randomness, although this is not a requisite of atypicality. What it means is that essentially we will use only second order properties of data.

Notice that the codelength (4.1) is not invariant to scaling. A natural approach to coding would be to do an invertible transformation of data, e.g. $\mathbf{y} = \mathbf{A}\mathbf{x}$. However, unless scaling issues is accounted for carefully, this can lead to false results.

Finally, one could ask if the real-valued problem should not be solvable with the binary approach of the previous section. To this there are two answers. First, if for example real a represented by 32 bit fixed point, then to detect atypicality of even a single sample a context length of 32 bits would be required, which has incredible complexity, since there are 2^{32} such contexts. Related to this, modeling data as real often leads to simpler models, and can capture more structure in the data, as mentioned above for signal processing methods.

4.1 Accurate Descriptive Length for Parametrized Models

There are several methods for accurate expression of MDL [79]. The most rigorous approach is normalized ML (NML) [79, 80], with a codelength given by

$$L(x^l) = -\log\left(\frac{f(x^l|\hat{\boldsymbol{\theta}}(x^l))}{\int f(x^l|\hat{\boldsymbol{\theta}}(x^l))dx^l}\right)$$

This is a strictly universal code in the sense that the regret [81] satisfies a minimax criterion. The issue is that the integral in the denominator is infinite for most interesting signal processing models, even iid Gaussian models. An elegant solution to this is the sequential NML (SNML) [82, 83], which instead of encoding the whole sequence encodes data predictively. The codelength of x_{n+1} given x^n is defined by

$$L(x_{n+1}|x^n) = -\log\left(\frac{f(x^{n+1}|\hat{\boldsymbol{\theta}}(x^{n+1}))}{\int f(\{x_{n+1}, x^n\}|\hat{\boldsymbol{\theta}}(x^{n+1}))dx_n}\right)$$

For many interesting signal processing models, the integral in the denominator is now finite. It does not satisfy the strict universality principle of the NML, but can be seen more as a heuristic method based on the principle of NML. Furthermore, for the first sample (at least) the coder cannot be used, so some other coder must be used. The main issue is that, as mentioned in [82, 83], the integral is often hard to evaluate, and there are still interesting signal processing models, e.g., auto-regressive (AR) models [84] where the integral is infinite.

For use in atypicality (it can of course also be used in other applications) we therefore introduce two new MDL methods, based on a common principle. Our starting point is Rissanen's [45] original predictive MDL

$$L(x^l) = -\sum_{i=0}^{l-1} \log f(x_{i+1}|\hat{\boldsymbol{\theta}}(x^i)) \tag{4.2}$$

The issue with this method is how to initialize the recursion. When $i = 0$, $\hat{\boldsymbol{\theta}}(x^i)$ is not defined. Rissanen suggests using a default pdf f_d to encode data until $\hat{\boldsymbol{\theta}}(x^i)$ is defined, so that $L(x^l) = -\sum_{i=1}^{l-1} \log f(x_{i+1}|\hat{\boldsymbol{\theta}}(x^i)) - \log f_d(x_1)$. In general, with more than one parameter, the default pdf might have to be used for more samples. The issue is that even when $\hat{\boldsymbol{\theta}}(x^i)$ *is* defined, the resulting codelength might be "poor." As an example consider a model $\mathcal{N}(0, \sigma^2)$ with only σ^2 unknown; then $\widehat{\sigma^2} = \frac{1}{n}\sum_{i=1}^{n} x_i^2$ for $n \geq 1$. Usually this gives a codelength close to (4.18). But for some sequences, this gives an extreme codelength, see Fig. 4.1. The issue is that for $n = 1$, $\widehat{\sigma^2}_{\mathrm{ML}} = x_1^2$, and if x_1 is small (and this is quite likely as the Gaussian has a big center mass), this gives a huge codelength for x_2, so large that it affects codelength for even long sequences. This is the issue our methodology addresses. The issue has been addressed in various ways previously [81]. One solution is to use a modified ML estimator [81]; but this is quite similar to using a prior distribution. Our idea is that rather than using the ML estimate for encoding as though it is the *actual* parameter value, we use it as an *uncertain* estimate of $\boldsymbol{\theta}$. We then take this uncertainty into account in the codelength. This is similar to the idea of using confidence intervals in statistical estimates [85]. In the following we introduce two methods using this general principle.

4.1.1 Normalized Likelihood Method (NLM)

Let the likelihood function of the model be $f(x^l|\boldsymbol{\theta})$. For a fixed x^l we can consider this as a "distribution" on $\boldsymbol{\theta}$; the ML estimate is of course the most likely value of this distribution. To account for uncertainty in the estimate, we can instead try use the total $f(x^l|\boldsymbol{\theta})$ to give a *distribution* on $\boldsymbol{\theta}$, and then use this for prediction. In general $f(x^l|\boldsymbol{\theta})$ is not a probability distribution as it does not integrate to 1 in $\boldsymbol{\theta}$. We can therefore normalize it to get a probability distribution

$$f_{x^l}(\boldsymbol{\theta}) = \frac{f(x^l|\boldsymbol{\theta})}{C(x^l)}; \quad C(x^l) = \int f(x^l|\boldsymbol{\theta}) d\boldsymbol{\theta} \tag{4.3}$$

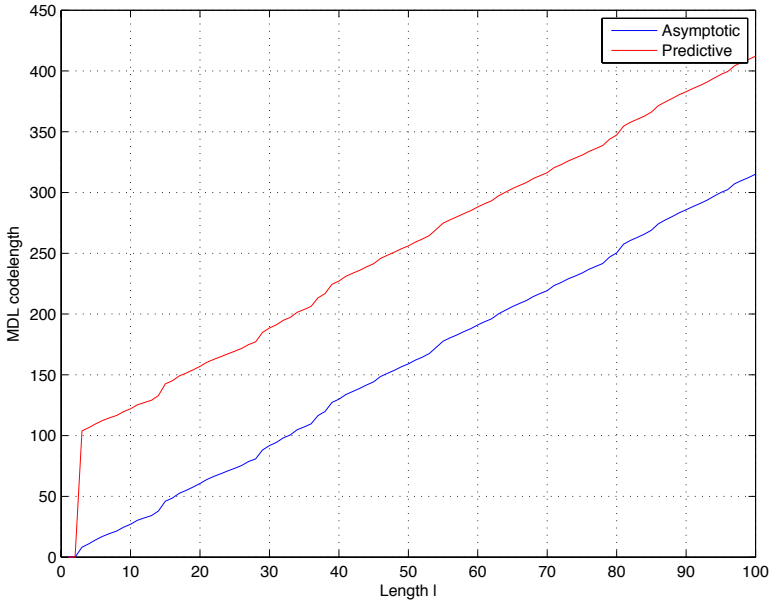

Figure 4.1: Predictive MDL for unknown variance.

if $\int f(x^l;\boldsymbol{\theta})d\boldsymbol{\theta}$ is finite. For comparison, the Bayes posteriori distribution is

$$f(\boldsymbol{\theta}|x^l) = \frac{f(x^l|\boldsymbol{\theta})f(\boldsymbol{\theta})}{\int f(x^l|\boldsymbol{\theta})f(\boldsymbol{\theta})d\boldsymbol{\theta}}$$

If the support Θ of $\boldsymbol{\theta}$ has finite area, (4.3) is just the Bayes predictor with uniform prior. If the support Θ of $\boldsymbol{\theta}$ does not have finite area, we can get (4.3) as a limiting case when we take the limit of uniform distributions on finite Θ_n that converge towards Θ. This is the same way the ML estimator can be seen as a MAP estimator with uniform prior [47]. One can reasonably argue that if we have no further information about $\boldsymbol{\theta}$, a uniform distribution seems reasonable, and has indeed been used for MDL [81] as well as universal source coding [39, Section 13.2]. What the Normalized Likelihood Method does is simply extend this to the case when there is no proper uniform prior for $\boldsymbol{\theta}$.

The method was actually implicitly mentioned as a remark by Rissanen in [86, Section 3.2], but to our knowledge was never further developed; here we introduce the method as a

practical method. From Rissanen we also know the coding distribution for x_n

$$f(x_{n+1}|x^n) = \int f(x_{n+1}|\boldsymbol{\theta}) f_{x^n}(\boldsymbol{\theta}) d\boldsymbol{\theta} = \frac{C\left(x^{n+1}\right)}{C\left(x^n\right)} \tag{4.4}$$

Let us assume $C(x^n)$ becomes finite for $n > 1$ (this is not always the case, often n needs to be larger). The total codelength can then be written as

$$\begin{aligned} L(x^l) &= \sum_{i=1}^{l-1} -\log f(x_{i+1}|x^i) - \log f_d(x_1) \\ &= -\log C(x^l) + \log C(x^2) - \log f_d(x_1) \end{aligned} \tag{4.5}$$

4.1.2 Sufficient Statistic Method (SSM)

The method of sufficient statistic is best explained through a simple example. Suppose our model is $\mathcal{N}(\mu, \sigma^2)$, with σ known. The average $\bar{x}_n$ is the ML estimate of μ at time n. We know that

$$\bar{x}_n = \mu + z, \quad z \sim \mathcal{N}\left(0, \frac{\sigma^2}{n}\right).$$

We can re-arrange this as

$$\mu = \bar{x}_n - z$$

Thus, *given* $\bar{x}_n$, we can think of μ as random $\mathcal{N}\left(\bar{x}_n, \frac{\sigma^2}{n}\right)$. Now

$$x_{n+1} = \mu + z_{n+1} \sim \mathcal{N}\left(\bar{x}_n, \sigma^2 + \frac{\sigma^2}{n}\right)$$

which we can use as a coding distribution for x_{n+1}. This compares to $\mathcal{N}\left(\bar{x}_n, \sigma^2\right)$ that we would use in traditional predictive MDL. Thus, we have taken into account that the estimate of μ is uncertain for n small. The idea of suddenly thinking of the non-random parameter μ as random might seem strange. However, the idea is very similar to the philosophical argument for confidence intervals [85].

In order to generalize this example to more complex models, we take the following approach. Suppose $\mathbf{t}(x^n)$ is a k-dimensional sufficient statistic for the k-dimensional $\boldsymbol{\theta} \in \Theta$; we know [47] that in many cases we can use $\mathbf{t}(x^n) = \hat{\boldsymbol{\theta}}$, the ML estimate of $\boldsymbol{\theta}$. Also suppose there exists some function $\mathbf{s}$ and a k-dimensional (vector) random variable $\mathbf{Y}$ *independent of* $\boldsymbol{\theta}$ so that

$$\mathbf{t}(x^n) = \mathbf{s}(\mathbf{Y}, \boldsymbol{\theta}). \tag{4.6}$$

We now assume that for every $(\mathbf{t}, \mathbf{Y})$ in their respective support there is a solution for $\boldsymbol{\theta} \in \Theta$ so that we can write

$$\boldsymbol{\theta} = \mathbf{r}(\mathbf{Y}, \mathbf{t}(x^n)).$$

The parameter $\boldsymbol{\theta}$ is now a random variable (assuming $\mathbf{r}$ is measurable, clearly) with a pdf $f_{x^n}(\boldsymbol{\theta})$. This then gives a distribution on x_{n+1}, i.e.,

$$f(x_{n+1}|x^n) = \int f(x_{n+1}|\boldsymbol{\theta}) f_{x^n}(\boldsymbol{\theta}) d\boldsymbol{\theta}$$

The intuition behind this approach is as follows. In order for (4.6) to have a (nice) solution, $\mathbf{t}$, $\boldsymbol{\theta}$ and $\mathbf{Y}$ need to have the same dimension, and therefore the observations need to be reduced to a k-dimensional statistic; in the motivating example for $n = 2$, we cannot solve $(x_1 = \mu + z_1, x_2 = \mu + z_2)$ with respect to μ. The statistic does need to be sufficient, since otherwise we could get arbitrary results; in the example we could solve $x_1 = \mu + z_1$ with respect to x_1 ignoring all other observations. That would give a completely different and non-sensible distribution on μ. On the other hand, we will show below that once we use a sufficient statistic, the distribution on $\boldsymbol{\theta}$ is essentially unique. Finally, it's important that (4.6) has a solution for every $(\mathbf{t}, \mathbf{Y})$. There might be models where for example for certain values of $\mathbf{t}$ one can say that certain values of $\mathbf{Y}$ are impossible. However, for fixed $\mathbf{t}$ the function $\mathbf{r}$ must be defined for all $\mathbf{Y}$ – otherwise $\boldsymbol{\theta}$ will not be a random variable.

The method has the following property

Theorem 6. The distribution of x_{n+1} is *invariant* to *arbitrary* parameter transformations.

Proof We let $\check{\boldsymbol{\theta}} = \mathbf{g}(\boldsymbol{\theta})$, where $\mathbf{g} : \mathbb{R}^k \to \mathbb{R}^k$ is one-to-one, if $\mathbf{t}$ is a sufficient for $\boldsymbol{\theta}$ it certainly is also sufficient for $\check{\boldsymbol{\theta}}$. But let us assume we use a different sufficient statistic $\tilde{\mathbf{t}}$. We necessarily have $\tilde{\mathbf{t}} = \tilde{\mathbf{g}}(\mathbf{t})$ where $\tilde{\mathbf{g}}$ is one-to-one on the support of the sufficient statistics (here we assume that the sufficient statistics are complete [47]). We can write

$$\begin{aligned}
\tilde{\mathbf{t}} &= \tilde{\mathbf{g}}(\mathbf{T}, \mathbf{g}^{-1}(\check{\boldsymbol{\theta}})) \\
\check{\boldsymbol{\theta}} &= \mathbf{g}(\mathbf{r}(\mathbf{T}, \tilde{\mathbf{g}}^{-1}(\tilde{\mathbf{t}})) \\
x_{n+1} &= \mathbf{h}(X, \mathbf{g}^{-1}(\check{\boldsymbol{\theta}})) = \mathbf{h}(X, \mathbf{r}(\mathbf{T}, \mathbf{t}))
\end{aligned}$$

Thus x_{n+1} has the same distribution as when we used $\mu, \mathbf{t}$. □

One concern is that way the method is described. Perhaps we could use different functions $\mathbf{s}$ and $\mathbf{r}$ and get a different result? In the following we will prove that the distribution of $\boldsymbol{\theta}$ is independent of which $\mathbf{s}$ and $\mathbf{r}$ are used. Let's start with one-dimensional case.

Theorem 7. Suppsoe $\theta \in \Theta \subset \mathbb{R}^k$ and assume that the support of t is independent of θ, and that it as well as Θ has finitely many connected components. Additionally assume that the function s in the model (4.6) is continuous and one-to-one in (Y, θ), and that the pdf of t, $s_t(t, \theta)$ is continuous in θ. Then the distribution of θ given by the sufficient statistic approach is unique.

Proof First notice that by [87] $F_\theta(t)$ is continuous in θ. It is well-known [39, 71] that $U = F_\theta(t)$ is uniformly distributed; we can also write $t = F_\theta^{-1}(U)$. Similarly, we can write $Y = F_Y^{-1}(V)$. Then

$$U = F_\theta(s(F_Y^{-1}(V), \theta)) = b_\theta(V)$$

is a one-to-one transformation of a uniform random variable to a uniform random variable. The transformation is continuous in V for $V \in F_Y(\text{supp } Y)$, and continuous for all θ due to the continuity of F_θ and the assumptions on s. For each connected component of $F_Y(\text{supp } Y)$, $b_\theta(V)$ is either identity or reversal, i.e., $\frac{\partial b_\theta(V)}{\partial V} = \pm 1$. Choose arbitrary $t_0 \in \text{int supp } t$ and $\theta_0 \in \text{int } \Theta$ and let $y_0 \in \text{supp } Y$ by so that $t_0 = s(y_0, \theta_0)$; as the support of t is assumed

independent of θ, we can find such a y_0. Because s is assumed to be one-to-one, for given (θ_0, t_0), the y_0 is the only y so that $t_0 = s(y, \theta_0)$. Let ϵ be so that $(\theta_0 - \epsilon, \theta_0 + \epsilon) \subset \Theta$ and so that for all $\theta \in (\theta_0 - \epsilon, \theta_0 + \epsilon)$ the unique solution y to $t_0 = s(y, \theta)$ satisfies $y \in \text{supp } Y$; this is possible as s is continuous and one-to-one (then the inverse is also continuous). Let $\mathcal{Y}$ be the set of solutions and let $\mathcal{V} = F_Y(\mathcal{Y})$. Then $b_\theta(V)$ is continuous on the connected set $(\theta_0 - \epsilon, \theta_0 + \epsilon) \times \mathcal{V}$. For a given θ and all $V \in \mathcal{V}$, $b_\theta(V) = cV + k$, where $c = \pm 1$ and k is a constant. Because $(\theta_0 - \epsilon, \theta_0 + \epsilon)$ is connected, c and k are independent of θ. Now $P((\theta_0 - \epsilon, \theta_0 + \epsilon); t_0) = P(\mathcal{Y}) = P(\mathcal{V})$. The latter probability can be written as

$$\begin{aligned} P((\theta_0-\epsilon, \theta_0+\epsilon); t_0) &= \left| b^{-1}_{\theta_0+\epsilon}(F_{\theta_0+\epsilon}(t_0)) - b^{-1}_{\theta_0-\epsilon}(F_{\theta_0-\epsilon}(t_0)) \right| \\ &= |F_{\theta_0+\epsilon}(t_0) - F_{\theta_0-\epsilon}(t_0)| \end{aligned}$$

Thus the probability $P((\theta_0 - \epsilon, \theta_0 + \epsilon))$ is independent of c, k, and since this is the only way the problem depends on s, the distribution of θ is independent of s. □

Now let's generalize it to higher dimension. It is well-known [39, 71] that if the random variable X has CDF F, then $U = F(X)$ has a uniform distribution (on $[0, 1]$). Equivalently, $X = F^{-1}(U)$ for some uniform random variable U. We need to generalize this to n dimensions. Recall that for a continuous random variable [71]

$$\begin{aligned} F_{i|i-1,\ldots,1}(x_i|x_{i-1}, \ldots x_1) &= \int_{-\infty}^{x_i} f(t|x_{i-1}, \ldots, x_1)dt \\ &= \frac{1}{f(x_{i-1}, \ldots, x_1)} \int_{-\infty}^{x_i} f(t, x_{i-1}, \ldots, x_1)dt \end{aligned}$$

whenever $f(x_{i-1}, \ldots, x_1) \neq 0$. As an example, let $n = 2$. Then the map $(X_1, X_2) \mapsto (F_1(X_1), F_{2|1}(X_2, X_1))$ is a map from $\mathbb{R}^2$ onto $[0, 1]^2$, and $(F_1(X_1), F_{2|1}(X_2, X_1))$ has uniform distribution on $[0, 1]^2$. Here $F_1(X_1)$ is continuous in X_1 and $F_{2|1}(X_2, X_1)$ is continuous in X_2.

We can write $X_1 = F_1^{-1}(U_1)$. For fixed x_1 we can also write $X_2 = F_{2|1}^{-1}(U_2|x_1)$ for those x_1 where $F_{2|1}$ is defined, and where the inverse function is only with respect to the parameter

before |. Then

$$\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} F_1^{-1}(U_1) \\ F_{2|1}^{-1}(U_2|F_1^{-1}(U_1)) \end{bmatrix} \triangleq \check{\mathbf{F}}^{-1}(U_1, U_2)$$

This gives the correct joint distribution on (X_1, X_2): the marginal distribution on X_1 is correct, and the conditional distribution of X_2 given X_1 is also correct, and this is sufficient. Clearly $\check{\mathbf{F}}^{-1}$ is not defined for all U_1, U_2; the relationship should be understood as being valid for almost all (X_1, X_2) and (U_1, U_2). We can now continue like this for $X_3, X_4, \ldots, X_n$. We will state this result as a lemma

Lemma 8. For any continuous random variable $\mathbf{X}$ there exists an n-dimensional uniform random variable $\mathbf{U}$, so that $\mathbf{X} = \check{\mathbf{F}}^{-1}(\mathbf{U})$. □

Theorem 9. Consider a model $\mathbf{t} = \mathbf{s}_1(\mathbf{Y}_1; \boldsymbol{\theta})$, with $\boldsymbol{\theta} = \mathbf{r}_1(\mathbf{Y}_1; \mathbf{t})$ and an alternative model $\mathbf{t} = \mathbf{s}_2(\mathbf{Y}_2; \boldsymbol{\theta})$, with $\boldsymbol{\theta} = \mathbf{r}_2(\mathbf{Y}_2; \mathbf{t})$. We make the following assumptions

1. The support of $\mathbf{t}$ is independent of $\boldsymbol{\theta}$ and its interior is connected.
2. The extended CDF $\check{\mathbf{F}}_i$ of $\mathbf{Y}_i$ is continuous and differentiable.
3. The function $\mathbf{Y}_i \mapsto \mathbf{s}_i(\mathbf{Y}_i; \boldsymbol{\theta})$ is one-to-one, continuous, and differentiable for fixed $\boldsymbol{\theta}$.

Then the distributions of $\boldsymbol{\theta}$ given by $\mathbf{r}_1$ and $\mathbf{r}_2$ are identical.

Proof By Lemma 8 write $\mathbf{Y}_1 = \mathbf{F}_1^{-1}(\mathbf{U}_1)$, $\mathbf{Y}_2 = \mathbf{F}_2^{-1}(\mathbf{U}_2)$. Let u be the k-dimensional uniform pdf, i.e, $u(\mathbf{x}) = 1$ for $\mathbf{x} \in [0,1]^k$ and 0 otherwise, and let $\mathbf{Y}_i = \mathbf{s}_i^{-1}(\mathbf{t}; \boldsymbol{\theta})$ denote the solution of $\mathbf{t} = \mathbf{s}_i(\mathbf{Y}_i; \boldsymbol{\theta})$ with respect to $\mathbf{Y}_i$, which is a well-defined due to assumption 3. We can then write the distribution of $\mathbf{t}$ in two ways as follows ([71]), due to the differentiability assumptions

$$\begin{aligned} f(\mathbf{t}; \boldsymbol{\theta}) &= u(\mathbf{F}_1(\mathbf{s}_1^{-1}(\mathbf{t}; \boldsymbol{\theta}))) \left| \frac{\partial \mathbf{F}_1(\mathbf{s}_1^{-1}(\mathbf{t}; \boldsymbol{\theta}))}{\partial \mathbf{t}} \right| \\ &= u(\mathbf{F}_2(\mathbf{s}_2^{-1}(\mathbf{t}; \boldsymbol{\theta}))) \left| \frac{\partial \mathbf{F}_2(\mathbf{s}_2^{-1}(\mathbf{t}; \boldsymbol{\theta}))}{\partial \mathbf{t}} \right| \end{aligned}$$

Due to assumption 1 we can then that conclude $\frac{\partial \mathbf{F}_1(\mathbf{s}_1^{-1}(\mathbf{t};\boldsymbol{\theta}))}{\partial \mathbf{t}} = \frac{\partial \mathbf{F}_2(\mathbf{s}_2^{-1}(\mathbf{t};\boldsymbol{\theta}))}{\partial \mathbf{t}}$, or

$$\mathbf{F}_1(\mathbf{s}_1^{-1}(\mathbf{t};\boldsymbol{\theta})) = \mathbf{F}_2(\mathbf{s}_2^{-1}(\mathbf{t};\boldsymbol{\theta})) + \mathbf{k}(\boldsymbol{\theta})$$

But both $\mathbf{F}_1$ and $\mathbf{F}_2$ have range $[0,1]^k$, and it follows that $\mathbf{k}(\boldsymbol{\theta}) = \mathbf{0}$. Therefore

$$\mathbf{t} = \mathbf{s}_1(\mathbf{F}_1^{-1}(\mathbf{U});\boldsymbol{\theta}) = \mathbf{s}_2(\mathbf{F}_2^{-1}(\mathbf{U});\boldsymbol{\theta})$$

if we then solve either for $\boldsymbol{\theta}$ as a function of $\mathbf{U}$ (for fixed $\mathbf{t}$), we therefore get exactly the same result, and therefore the same distribution. □

The assumptions of Theorem 9 are very strong, but we believe they are far from necessary. In Theorem 7 we proved uniqueness in the one-dimensional case under much weaker assumptions (e.g., no differentiability assumptions), but that proof is not easy to generalize to higher dimensions. Loosely one can say that the distribution of $\boldsymbol{\theta}$ is unique when the model is reasonably nice, without being able to specify the minimum conditions for "niceness."

Corollary 10. Let $\mathbf{t}_1(x^n)$ and $\mathbf{t}_2(x^n)$ be *equivalent* sufficient statistic for $\boldsymbol{\theta}$. Then the distribution on $\boldsymbol{\theta}$ given by the sufficient statistic approach is the same for $\mathbf{t}_1$ and $\mathbf{t}_2$.

Proof We have $\mathbf{t}_1 = \mathbf{s}_1(\mathbf{Y}_1, \boldsymbol{\theta})$ and $\mathbf{t}_2 = \mathbf{s}_2(\mathbf{Y}_2, \boldsymbol{\theta})$. By assumption, there exists a one-to-one map a so that $\mathbf{t}_1 = a(\mathbf{t}_2)$, thus $\mathbf{t}_1 = a(\mathbf{s}_2(\mathbf{Y}_2, \boldsymbol{\theta}))$. Since the distribution of $\boldsymbol{\theta}$ is independent of how the problem is stated, $\mathbf{t}_1$ and $\mathbf{t}_2$ gives the same distribution on $\boldsymbol{\theta}$. □

We will compare the methods for the model of Fig. 4.1. Assume our model is $\mathcal{N}(0, \sigma^2)$ with σ unknown. The likelihood function is $f(x^n|\sigma^2) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^n x_i^2\right)$. For $n = 1$ we have $\int_0^\infty f(x^n|\sigma^2)d\sigma^2 = \infty$, but for $n \geq 2$

$$C\left(x^n\right) = \int f(x^n|\sigma^2)d\sigma^2 = \frac{1}{\pi^{\frac{n}{2}}2}\frac{\Gamma\left(\frac{n-2}{2}\right)}{\left[n\widehat{\sigma^2}_n\right]^{\frac{n-2}{2}}}$$

then

$$f_{\mathrm{nlm}}(x_{n+1}|x^n) = \frac{\Gamma\left(\frac{n-1}{2}\right)}{\sqrt{\pi}\Gamma\left(\frac{n-2}{2}\right)} \frac{\left[n\widehat{\sigma^2}_n\right]^{\frac{n-2}{2}}}{\left[(n+1)\,\widehat{\sigma^2}_{n+1}\right]^{\frac{n-1}{2}}}$$

where $\widehat{\sigma^2}_n = \frac{1}{n}\sum_{i=1}^{n} x_i^2$. Thus, for coding, the two first sample would be encoded with the default distribution, and after that the above distribution is used. Now applying the sufficient statistic method we have $z = \frac{n}{\sigma^2}\widehat{\sigma^2}_n \sim \chi^2_{(n)}$, so

$$f_{\mathbf{x^n}}(\sigma^2) = \frac{\left[n\widehat{\sigma^2}_n\right]^{\frac{n}{2}}}{2^{\frac{n}{2}}\Gamma\left(\frac{n}{2}\right)(\sigma^2)^{\frac{n+2}{2}}} \exp\left\{-\frac{n}{2\sigma^2}\widehat{\sigma^2}_n\right\}$$

now we have

$$\begin{aligned} f_{\mathrm{ssm}}(x_{n+1}|x^n) &= \int f(x_{n+1}|\sigma^2) f_{x^n}(\sigma^2) d\sigma^2 \\ &= \frac{\Gamma\left(\frac{n+1}{2}\right)}{\sqrt{\pi}\Gamma\left(\frac{n}{2}\right)} \frac{\left[n\widehat{\sigma^2}_n\right]^{\frac{n}{2}}}{\left[(n+1)\,\widehat{\sigma^2}_{n+1}\right]^{\frac{n+1}{2}}} \end{aligned} \tag{4.7}$$

The normalized likelihood method and the sufficient statistic method give almost the same result. The sufficient statistic method, though, as outlined above, has the advantage that it is invariant to arbitrary parameter transformations, and then the added advantage that it works for $n \geq 1$ rather than $n \geq 3$. This can be quite significant, as we want to depend as little as possible on the default pdf, which is highly subjective.

It turns out that SNML gives exactly the same codelength as (4.7). This is true for many models, e.g., examples IV.B and IV.C in [82]. It is not clear why SNML and SSM should give the same codelength, as the principles behind are quite different; but it is in some sense encouraging, as it shows that this is perhaps the "right" codelength. However, they don't give the same codelength for all models (see Section 4.1.3.1), and there are models where

one is defined, but not the other. So, they are not different paths to the same MDL.

For comparison, the ordinary predictive MDL is

$$f(x_{n+1}|x^n) = \frac{1}{\sqrt{2\pi\widehat{\sigma^2}_n}} \exp\left(-\frac{1}{2\widehat{\sigma^2}_n} x_{n+1}^2\right) \tag{4.8}$$

which is of a completely different form. To understand the difference, consider the codelength for x_2

$$\begin{aligned} L(x_2) &= \log\left(\frac{x_1^2 + x_2^2}{|x_1|}\right) + \log\left(\frac{\sqrt{\pi}\Gamma(\frac{1}{2})}{\Gamma(1)}\right) && \text{suff. stat.} \\ L(x_2) &= \frac{1}{2}\log\left(2\pi x_1^2\right) + \frac{x_2^2}{x_1^2} && \text{predictiv MDL} \end{aligned}$$

At can be seen that if x_1 is small and x_2 is large, the codelength for x_2 is going to be large. But in the sufficient statistic method this is strongly attenuated due to the log in front of the ratio. Fig. 4.2 shows this quantitatively in the redundancy sense (difference between the codelength using true and estimated distributions). As can be seen, the CDF of the ordinary predictive MDL redundancy has a long tail which is what results in the behavior in Fig. 4.1, and this is taken care of by our method.

Another advantage of the sufficient statistic/normalized likelihood method is that they can be calculated for a whole block of data: while in principle recursive, the implementation does not have to be recursive. For the NLM this is explicit from (4.5), and for the SSM we get from (4.7)

$$L(x^n) = -\log\left(f_d(x_1)\right) + \log\left(\frac{\left(n\widehat{\sigma^2}_n\right)^{\frac{n}{2}}}{|x_1|}\right) - \log\frac{\Gamma(\frac{n}{2})}{\sqrt{\pi}\Gamma(\frac{n-1}{2})}$$

While we have no proof that a block-based implementation is always possible for SSM, it turns out to be the true in all the cases we have analyzed. On the other hand, the ordinary predictive MDL (4.8) does not have an obvious block-based implementation. It is clear that

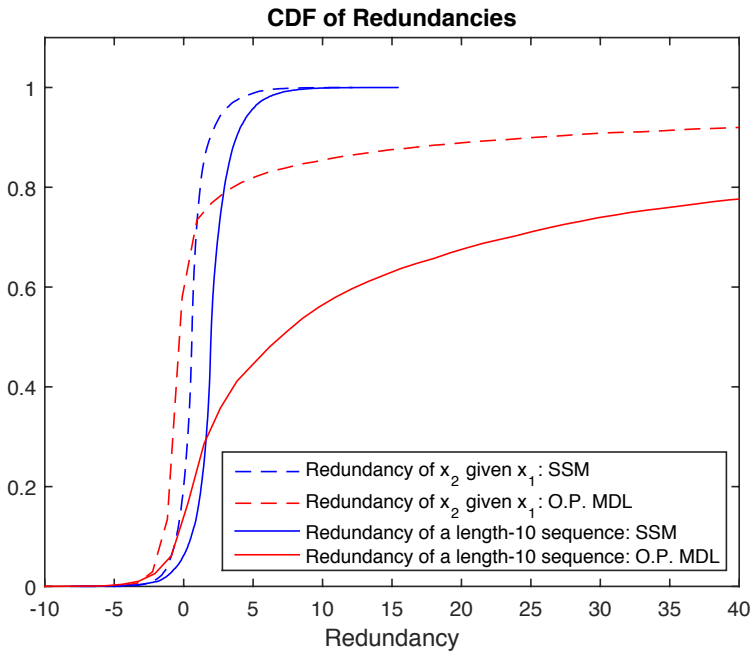

Figure 4.2: Redundancy comparison between ordinary predictive MDL and our proposed sufficient statistic method for $\mu = 0$ and $\sigma^2 = 4$.

a block-based implementation is of much lower complexity, unless variable length sequences are analyzed, as it is for example the case with finding atypical subsequences.

4.1.3 Scalar Signal Processing Methods

In the following we will derive MDL for various scalar signal processing methods. We can take inspiration from signal processing methods generally used for source coding, such as linear prediction and wavelets; however, the methods have to be modified for MDL, as we use lossless coding, not lossy coding. All proofs are in Appendix.

4.1.3.1 Iid Gaussian Case

A natural extension of the examples considered in Section 4.1.2 is $x_n \sim \mathcal{N}(\mu, \sigma^2)$ with both μ and σ^2 unknown. Define $\hat{\mu}_n = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $S_n^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \hat{\mu}_n)^2$. Then the sufficient

statistic method is

$$f(x_{n+1}|x^n) = \sqrt{\frac{n}{\pi\,(n+1)}}\frac{\Gamma\left(\frac{n}{2}\right)}{\Gamma\left(\frac{n-1}{2}\right)}\frac{\left[(n-1)\,S_n^2\right]^{\frac{n-1}{2}}}{\left[nS_{n+1}^2\right]^{\frac{n}{2}}} \tag{4.9}$$

This is a special case of the vector Gaussian model considered later, so we will not provide a proof.

Linear Transformations

The iid Gaussian case is a fundamental building block for other MDL methods. The idea is to find a linear transformation so that we can model the result as iid, and then use the iid Gaussian MDL. For example, in the vector case, suppose $\mathbf{x}_n \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is (temporally) iid, and let $\mathbf{y}_n = \mathbf{A}\mathbf{x}_n \sim N(\mathbf{A}\boldsymbol{\mu}, \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T)$. If we then *assume* that $\mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T$ is diagonal, we can use the iid Gaussian MDL on each component. Similarly, in the scalar case, we can use a filter instead of a matrix. Because codelength is not scaling invariant, we need to require $\mathbf{A}$ to be orthonormal: for any input we than have $\mathbf{y}_n^T\mathbf{y}_n = \mathbf{x}_n^T\mathbf{A}^T\mathbf{A}\mathbf{x}_n = \mathbf{x}_n^T\mathbf{x}_n$, and in particular $E[\mathbf{y}_n^T\mathbf{y}_n] = E[\mathbf{x}_n^T\mathbf{x}_n]$ independent of the actual $\boldsymbol{\Sigma}$. We will see this approach in several cases in the following. Of course, we do not *know* that the result is iid Gaussian, it is a model assumption, and the idea is that if this is a better approximation than the typical model, we get a shorter codelength.

4.1.3.2 Linear Prediction

Linear prediction is a fundamental to random processes. Write

$$\hat{x}_{n+1|x^n} = \sum_{k=0}^{\infty} w_k x_{n-k}$$

$$e_{n+1} = x_{n+1} - \hat{x}_{n+1|x^n}$$

Then for most stationary random processes the resulting random process $\{e_n\}$ is uncorrelated, and hence in the Gaussian case, iid, by the Wold decomposition [71]; denote by τ the power of $\{e_n\}$. It is therefore a widely used method for source coding, e.g., [78]. In practical coding, a finite prediction order M is used,

$$\hat{x}_{n+1|x^n} = \sum_{k=1}^{M} w_k x_{n-k+1}, \quad n \geq M$$

Consider the simplest case with $M = 1$: there are two unknown parameters (w_1, τ). However, the minimal sufficient statistic has dimension three [88]: $\left(\sum_{k=1}^{n} x_k^2, \sum_{k=1}^{n-1} x_k^2, \sum_{k=2}^{n} x_k x_{k-1}\right)$. Therefore, we cannot use SSM; and even if we could, the distribution of the sufficient statistic is not known in closed form [88]. We therefore turn to the NLM.

We assume that $e_{n+1} = x_{n+1} - \hat{x}_{n+1|x^n}$ is iid normally distributed with zero mean and variance τ,

$$f(x^n|\tau, \mathbf{w}) = \frac{1}{(2\pi\tau)^{(n-M)/2}} \exp\left(-\frac{1}{2\tau} \sum_{i=M+1}^{n} \left[x_i - \sum_{k=1}^{M} w_k x_{i-k}\right]^2\right) \tag{4.10}$$

Define

$$\hat{r}_{(n)}(k) = \sum_{i=M+1}^{n} x_i x_{i-k}$$

so that we can write

$$\begin{aligned}
\sum_{i=M+1}^{n} e_i^2 &= \sum_{i=M+1}^{n} \left[x_i - \sum_{k=1}^{M} w_k x_{i-k} \right]^2 \\
&= \sum_{i=M+1}^{n} x_i^2 - 2 \sum_{i=M+1}^{n} \sum_{k=1}^{M} w_k x_i x_{i-k} + \sum_{i=M+1}^{n} \sum_{k=1}^{M} \sum_{p=1}^{M} w_k w_p x_{i-k} x_{i-p} \\
&= \hat{r}_{(n)}(0) - 2 \sum_{k=1}^{M} w_k \sum_{i=M+1}^{n} x_i x_{i-k} + \sum_{k=1}^{M} \sum_{p=1}^{M} w_k w_p \sum_{i=M+1}^{n} x_{i-k} x_{i-p} \\
&= \hat{r}_{(n)}(0) - 2 \sum_{k=1}^{M} w_k \hat{r}_{(n)}(k) + \sum_{k=1}^{M} \sum_{p=1}^{M} w_k w_p \sum_{i=M+1}^{n} x_{i-k} x_{i-p} \\
&= \hat{r}_{(n)}(0) - 2 \mathbf{w}^T \mathbf{p}_{(n)} + \mathbf{w}^T R_{(n)}^{(M)} \mathbf{w}
\end{aligned}$$

where $\mathbf{w}^T = [w_1 \; w_2 \; \cdots \; w_M]$, $\mathbf{p}_{(n)}^T = [\hat{r}_{(n)}(1) \; \hat{r}_{(n)}(2) \; \cdots \; \hat{r}_{(n)}(M)]$,

$$R_{(n)}^{(M)} = \sum_{i=M+1}^{n} \mathbf{x}_{i-M}^{i-1} \left(\mathbf{x}_{i-M}^{i-1} \right)^T \tag{4.11}$$

and $x_{i-M}^{i-1} = [x_{i-1}, x_{i-2}, \ldots, x_{i-M}]$. Thus

$$f(x^n|\tau, \mathbf{w}) = \frac{1}{(2\pi\tau)^{(n-M)/2}} \exp\left(-\frac{1}{2\tau} \left[\hat{r}_{(n)}(0) - 2\mathbf{w}^T \mathbf{p}_{(n)} + \mathbf{w}^T R_{(n)}^{(M)} \mathbf{w} \right] \right)$$

giving (see Appendix A.4)

$$C(x^n) = \frac{1}{2\left(\pi\right)^{\frac{n-2M}{2}} \sqrt{\det\left(R_{(n)}\right)}} \frac{\Gamma\left(\frac{n-2M-2}{2}\right)}{\left(\hat{\tau}_{(n)}^{(M)}\right)^{\frac{n-2M-2}{2}}}$$

and

$$f_M(x_{n+1}|x^n) = \sqrt{\frac{\det\left(R_{(n)}^{(M)}\right)}{\det\left(R_{(n+1)}^{(M)}\right)}} \frac{\Gamma\left(\frac{n-2M-1}{2}\right)}{\Gamma\left(\frac{n-2M-2}{2}\right)} \frac{1}{\sqrt{\pi}} \frac{\left(\hat{\tau}_{(n)}^{(M)}\right)^{\frac{n-2M-2}{2}}}{\left(\hat{\tau}_{(n+1)}^{(M)}\right)^{\frac{n-2M-1}{2}}} \tag{4.12}$$

with $\hat{\tau}_{(n)}^{(M)} = \hat{r}_{(n)}(0) - \mathbf{p}_{(n)}^T R_{(n)}^{-1} \mathbf{p}_{(n)}$.

We will next discuss implementation aspects of the method. There are two cases: block-based implementation and recursive implementation. For the former, we can use (4.5), using $C(x^n)$ for the smallest n where its defined, see below. For recursive implementation, we can use recursive least-squares (RLS) [89] for updating the quantities as follows

$$\left(R_{(n+1)}^{(M)}\right)^{-1} = \left(R_{(n)}^{(M)}\right)^{-1} - \frac{\left(R_{(n)}^{(M)}\right)^{-1} x_{n-M}^{n-1} \left(x_{n-M}^{n-1}\right)^T \left(R_{(n)}^{(M)}\right)^{-1}}{1 + \left(x_{n-M}^{n-1}\right)^T \left(R_{(n)}^{(M)}\right)^{-1} x_{n-M}^{n-1}}$$

$$\det\left(R_{(n+1)}^{(M)}\right) = \det\left(R_{(n)}^{(M)}\right) \left(1 + \left(x_{n-M}^{n-1}\right)^T \left(R_{(n)}^{(M)}\right)^{-1} x_{n-M}^{n-1}\right)$$

by the matrix inversion lemma and the determinant lemma [90]. Thus, the complexity of calculating the sufficient statistic MDL is no more than any other linear predictor[1]

The equation (4.12) is defined for $n \geq 2M+2$: the vector x_{i-M}^{i-1} is defined for $i \geq M+1$, and $R_{(n)}^{(M)}$ defined by (4.11) becomes full rank when the sum contains M terms. This startup time can be reduced by assuming that $x_i = 0$ for $i < 1$; Then x_{i-M}^{i-1} is defined for $i \geq 2$, and (4.12) for $n > M+1$. However, with that initialization, the assumption that the e_i are iid in (4.10) seems not quite reasonable, though in some applications of atypicality the zero initial conditions could be reasonable. In any case, before the order M linear predictor becomes defined, the data needs to be encoded with other methods. Since in atypicality we are not seeking to determine the model of data, just if a different model than the typical is better, we encode data with lower order linear predictors until the order M linear predictor becomes

[1]Notice that since (4.11) is not at circulant matrix, Levinson-Durbin cannot be used for order-recursive calculation. Each order linear predictor has to be calculated separately.

defined. So, the first sample is encoded with the default pdf. The second and third samples are encoded with the iid unknown variance coder (4.7)[2]. Then the order 1 linear predictor takes over, and so on.

4.1.3.3 Filterbanks and Wavelets

After introduction of quadrature-mirror filters and then conjugate mirror filters, many multirate systems use filterbanks for signal compression, in which first the original signal is separated into subband signals with non-overlapping frequency bands and then the subband signals are compressed. This system of subband coding ensure better compression than the direct compression of original signal, since it allocates various number of bits to each subband signal resulting in lower average bit rate per sample [91, 92]. With almost the same intuition, we want to encode each subband signal separately. Note that only in this section, for the sake of notation, we use [.] to refer to sample index, so for example $x[n]$ refers to n-th sample of x.

Suppose $x\,[n] \sim \mathcal{N}(0, \sigma^2)$ is the iid input of a filterbank, obviously in MDL framework encoding the input is at least as good as encoding the outputs of a filterbank with any depth and structure since $x\,[n]$ have almost constant integrated power at different frequency bands. Now if the input of the filterbank $\hat{x}[n]$ is a filtered version of the input (e.g., output of an AR process), then $\hat{x}[n]$ will have different integrated power at different frequency bands, therefore it is likely to achieve fewer codelength by encoding the outputs in some structures of a filterbank that allows encoding of different frequency bands with distinct power separately. This claim will be further analyzed in section 4.1.3.3.

Filterbank Structure

Unlike many signal and image compression schemes in which the outputs of low-pass filters are of interest since they are approximated signals containing most of the information at

[2]There is no issue in encoding some samples with SSM and others with NLM

various levels, MDL and atypicality frameworks suggest that in a filterbank of depth D, codelength in all possible combinations of outputs in depths $d \leq D$ must be calculated and the structure that gives the minimum codelength should be chosen. Let's call the set of all possible binary tree (not necessarily symmetric) having depth not larger than D the *model class* $\mathcal{C}_D$. Also define $\mathbf{y}_D(\mathcal{S})$ as the outputs of the filterbank with structure model $\mathcal{S} \in \mathcal{C}_D$ and let $\mathbf{Y}_D$ be the set of all outputs for each $\mathcal{S} \in \mathcal{C}_D$. Additionally define $\mathbf{f}_D(\mathcal{S})$ as the signals that are passed to filters in the filterbank to generate outputs in $\mathbf{y}_D(\mathcal{S})$ and let $\mathbf{F}_D$ be the set of all signals that are used as input of filters for each $\mathbf{y}_D \in \mathbf{Y}_D$. The first step is to find the total number of possible trees in $\mathcal{C}_D$, i.e., $|\mathcal{C}_D|$. This process can be shown through an example. Assume a depth-one filterbank with input signal $x[n]$ and output signals $y_1[n]$ and $y_2[n]$, obviously there is only one possible combination of outputs: pair of $\mathbf{Y}_D = \{(y_1, y_2)\}$, therefore $\mathbf{F}_D = \{(x)\}$ and $|\mathcal{C}_D| = 1$. Now assume the case with depth two in Fig. 4.3. The total number of possible output combination is $|\mathcal{C}_D| = 2^2$, $\mathbf{Y}_D = \{(y_3, y_4, y_5, y_6), (y_2, y_3, y_4), (y_1, y_5, y_6), (y_1, y_2)\}$ and $\mathbf{F}_D = \{(x, y_1, y_2), (x, y_1), (x, y_2), (x)\}$. It can be shown that the total number of possible output combinations $r(d)$ at depth $d \geq 2$ can be calculated using the following recursive equation

$$r(d) = (r(d-1) + 1)^2$$

where the initial condition is $r(1) = 1$. Finally given the maximum depth D we have $|\mathcal{C}_D| = r(D)$. Generally, given any maximum depth D, MDL and atypicality framework ensure finding a structure $\mathcal{S} \in \mathcal{C}_D$ that gives the minimum codelength. Later we'll see searching for an optimum structure can be done efficiently.

Filter choices and transient effect

As it was earlier alluded, for any maximum depth that we consider for a filterbank, all possible structures should be considered and the one with minimum codelength should be chosen; however, in practice there is a limitation on the maximum depth due to the FIR

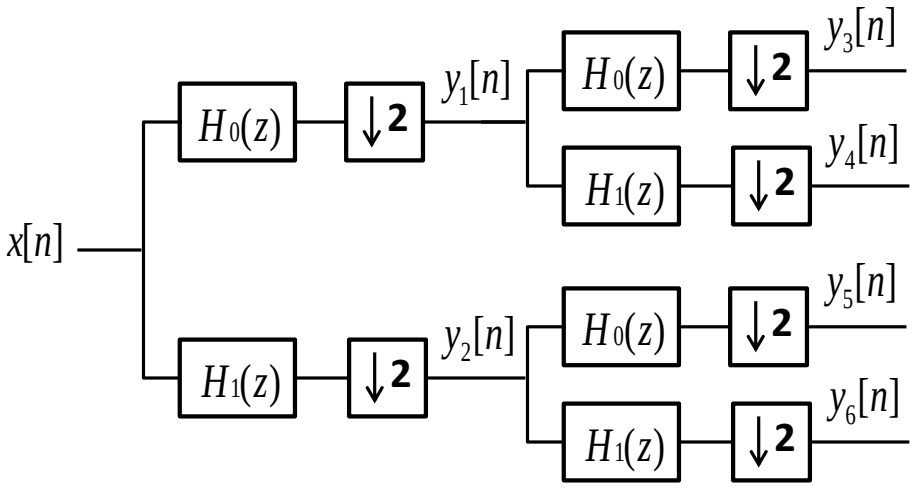

Figure 4.3: A depth-two filterbank with equal passband widths.

filters that we use. Suppose we have a causal FIR filter with length M and a causal input of length $N > M$. Then the causal output of the filter has a transient response for $n < M$ since input has partially engaged the filter, and a steady-state response for $n \geq M$. Now if N is not large, the transient response will be problematic. This issue is more severe in a deep filterbank structure since in each level not only transient response of filters are critical, but also downs-samplers decrease the number of samples. Another issue here is that even with assumption of iid for input of filters, the transient response is not iid and should be treated differently. Ergo the length of FIR filter that we choose and also the depth of the filterbanks depend highly on the number of input samples at hand. As a result, perfect reconstruction wavelet filters with small M are of interest. We also consider only orthogonal filterbanks. In next subsection will discuss how transient response and steady-state response is treated in our coding procedure.

Atypical coding

Suppose we have filterbank with maximum depth D and the corresponding model class $\mathcal{C}_D$ with $|\mathcal{C}_D|$ possible binary tree structures, also assume we use a length-M perfect reconstruction two-channel wavelet filters. Let the filterbank input be a length-N iid Gaussian with

zero mean and unknown variance $x[n] \sim \mathcal{N}(0, \sigma^2)$. For encoding $x^N = x[1], \ldots, x[N]$ we use the predictive distribution of equation (4.7)

$$f\left(x^N\right) = f_d\left(x[1]\right) \frac{\Gamma(\frac{N}{2})}{\pi^{\frac{N}{2}}} \frac{\left(\widehat{\sigma^2}_1\right)^{\frac{1}{2}}}{\left(N\widehat{\sigma^2}_N\right)^{\frac{N}{2}}}$$

where f_d is some default pdf and $\widehat{\sigma^2}_N = \frac{1}{N}\sum_{i=1}^{N} x[i]^2$. In order to ease the notation let's define the codelength function $L_G\left(x^N\right) = -\log f\left(x^N\right) = -\log f_d\left(x[1]\right) - \log|x[1]|\Gamma(\frac{N}{2}) + \frac{N}{2}\log \pi N\widehat{\sigma^2}_N$. To start the coding procedure, first assume we have a depth-one filterbanks, i.e., $D = 1$. As we talked in previous subsection, the transient response of filters are not iid and therefore in order to avoid transient response, we use the first M samples of input as initial condition of filters, hence the filterbank outputs (after downsamplers) $y_1[n]$ and $y_2[n]$ have $(N-M)/2$ sample each, both iid Gaussian with zero mean and unknown variances. In addition to the outputs that are encoded separately using the predictive distribution of equation (4.7), we should also encode the first M samples of input $x[n]$ separately using the same equation (Note that these M samples are iid since $x[n]$ is iid). Consequently $y_1^{\frac{N-M}{2}}$, $y_2^{\frac{N-M}{2}}$ and x^M should be encoded separately, ergo the total codelength of of a depth-one filterbanks is

$$L = L_G\left(y_1^{\frac{N-M}{2}}\right) + L_G\left(y_2^{\frac{N-M}{2}}\right) + L_G\left(x^M\right) \tag{4.13}$$

Now before we talk about deeper filterbanks, lets analyze the above equation and show it can reduce the codelength, basically we have to compare $L_G\left(x^N\right)$ and $L_G\left(y_1^{\frac{N-M}{2}}\right) + L_G\left(y_2^{\frac{N-M}{2}}\right) + L_G\left(x^M\right)$. This comparison can be reduced to

$$\frac{N}{2}\log\widehat{\sigma^2}_N \lessgtr \frac{N-M}{4}\log\left(\widehat{\sigma^2}_{1,\frac{N-M}{2}}\widehat{\sigma^2}_{2,\frac{N-M}{2}}\right) + \frac{M}{2}\log\widehat{\sigma^2}_M + c_1$$

without loss of generality suppose $N \gg M$, therefore

$$\frac{N}{2}\log\widehat{\sigma^2}_N \lessgtr \frac{N}{4}\log\frac{\widehat{\sigma^2}_{1,\frac{N}{2}}\widehat{\sigma^2}_{2,\frac{N}{2}}}{c_2}$$

so based on $\widehat{\sigma^2}_N \lessgtr \sqrt{\frac{\widehat{\sigma^2}_{1,\frac{N}{2}}\widehat{\sigma^2}_{2,\frac{N}{2}}}{c_2}}$ it is possible to reduce the codelength using filterbank decomposition.

Now let's move on to calculate the total codelength for a filterbank with any depth. In general, for all $d \leq D$ and all filterbank structures $\mathcal{S} \in \mathcal{C}_d$, we assume each output in the set $\mathbf{y}_d(\mathcal{S})$ is iid Gaussian with zero mean and unknown variance and we encode each $y_i \in \mathbf{y}_d(\mathcal{S})$ separately using the predictive distribution of equation (4.7), also for each signal in $\mathbf{f}_D(\mathcal{S})$, the first M samples are iid Gaussian with zero mean and unknown variances and are encoded separately using the same equation. Hence the total atypical code length is then given by

$$L = \min_{d\leq D}\min_{\mathcal{S}\in\mathcal{C}_d}\left\{\sum_{y_i\in\mathbf{y}_d(\mathcal{S})} L_G\left(y_i^{\cdots}\right) + \sum_{y_i\in\mathbf{f}_D(\mathcal{S})} L_G\left(y_i^M\right) + \log d + \log^*|\mathcal{C}_d|\right\}$$

where $y_i^{\cdots}$ refers to all samples of $y_i[n]$ and the terms $\log d + \log^*|\mathcal{C}_d|$ is to tell decoder which depth and structure for filterbank is used.

Now one issue that should be discussed here is that given a maximum depth, how can we efficiently search for the best structure that gives the smallest codelength. Suppose maximum depth is D, starting from depth-one filterbank if the total codelength of the outputs in equation (4.13) is not less than $L_G\left(x^N\right)$, we stop since frequency decomposition won't reduce the codelength. On the other hand, if depth-one filterbank reduce the total codelength, then we have to add another layer to the filterbank and do the same test for $y_1[n]$ and $y_2[n]$ which are the inputs of second layer, i.e., based on Fig. 4.3 investigate whether: (1) the decomposition of $y_1[n]$ into $y_3[n]$ and $y_4[n]$ reduce the codelength, (2) the decomposition of $y_2[n]$ into $y_5[n]$ and $y_6[n]$ reduce the codelength or (3) both. This procedure should be done until we get to a point that no further decomposition reduce codelength or we reach

the maximum depth.

Another question that should be answered is that can a decoder reconstruct the main signal? The only difference between our coding procedure and a perfect-reconstruction filterbank is in the way we avoid transient response of filters by sending samples that produce the transient response directly. Since both encoder and decoders know filters, then a decoder can produce transient response using the samples that they received directly, hence a decoder have both transient and steady-state responses and therefore can reconstruct the main signal.

4.1.4 Vector Signal Processing Methods

4.1.4.1 Vector Gaussian Case with unknown mean

First assume $\boldsymbol{\mu}$ is unknown but Σ is given. We define $\text{etr}\left(\cdots\right) = \exp\left(\text{trace}\left(\cdots\right)\right)$ and we have

$$f\left(\mathbf{x}^n|\boldsymbol{\mu}\right) = \frac{1}{\sqrt{\left(2\pi\right)^{kn}\det\left(\Sigma\right)^n}} \exp\left\{-\frac{1}{2}\sum_{i=1}^{n}\left(\mathbf{x}_i-\boldsymbol{\mu}\right)^T\Sigma^{-1}\left(\mathbf{x}_i-\boldsymbol{\mu}\right)\right\}$$

now we first want to use normalized likelihood method. By defining $\hat{\boldsymbol{\mu}}_n = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$ and $\hat{\Sigma}_n = \sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i^T$ (note that $\hat{\Sigma}_n$ is not the estimation of Σ) we have

$$\begin{aligned}
C\left(\mathbf{x}^n\right) &= \int f\left(\mathbf{x}^n|\boldsymbol{\mu}\right)d\boldsymbol{\mu} \\
&= \frac{1}{\sqrt{\left(2\pi\right)^{kn}\det\left(\Sigma\right)^n}} \exp\left\{-\frac{1}{2}\sum_{i=1}^{n}\mathbf{x}_i^T\Sigma^{-1}\mathbf{x}_i\right\}\int \exp\left\{-\frac{n}{2}\boldsymbol{\mu}^T\Sigma^{-1}\boldsymbol{\mu}+n\hat{\boldsymbol{\mu}}_n^T\Sigma^{-1}\boldsymbol{\mu}\right\}d\boldsymbol{\mu} \\
&= C\exp\left\{-\frac{1}{2}\sum_{i=1}^{n}\left(\mathbf{x}_i^T\Sigma^{-1}\mathbf{x}_i-\hat{\boldsymbol{\mu}}_n^T\Sigma^{-1}\hat{\boldsymbol{\mu}}_n\right)\right\} \\
&= C\text{etr}\left\{-\frac{1}{2}\left(\hat{\Sigma}_n-n\hat{\boldsymbol{\mu}}_n\hat{\boldsymbol{\mu}}_n^T\right)\Sigma^{-1}\right\}
\end{aligned}$$

where $C = \frac{1}{\sqrt{(2\pi)^{k(n-1)} n^k \det(\Sigma)^{n-1}}}$ hence we can write

$$
\begin{aligned}
f\left(\mathbf{x}_{n+1}|\mathbf{x}^n\right) &= \frac{C\left(\mathbf{x}^{n+1}\right)}{C\left(\mathbf{x}^n\right)} \\
&= \sqrt{\left(\frac{n}{n+1}\right)^k} \frac{1}{\sqrt{(2\pi)^k \det\left(\Sigma\right)}} \frac{\operatorname{etr}\left\{-\frac{1}{2}\left(\hat{\Sigma}_{n+1} - (n+1)\,\hat{\boldsymbol{\mu}}_{n+1}\hat{\boldsymbol{\mu}}_{n+1}^T\right)\Sigma^{-1}\right\}}{\operatorname{etr}\left\{-\frac{1}{2}\left(\hat{\Sigma}_n - n\hat{\boldsymbol{\mu}}_n\hat{\boldsymbol{\mu}}_n^T\right)\Sigma^{-1}\right\}}
\end{aligned} \tag{4.14}
$$

A sufficient statistic for $\boldsymbol{\mu}$ is $\hat{\boldsymbol{\mu}}_n = \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i \sim \mathcal{N}\left(\boldsymbol{\mu}, \frac{1}{n}\Sigma\right)$. Thus we can write

$$
\begin{aligned}
\hat{\boldsymbol{\mu}}_n &= \boldsymbol{\mu} + \mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}\left(0, \frac{1}{n}\Sigma\right) \\
\boldsymbol{\mu} &= \hat{\boldsymbol{\mu}}_n - \mathbf{z} \sim \mathcal{N}\left(\hat{\boldsymbol{\mu}}_n, \frac{1}{n}\Sigma\right) \\
\mathbf{x}_{n+1} &= \boldsymbol{\mu} + \mathbf{z}_n \sim \mathcal{N}\left(\hat{\boldsymbol{\mu}}_n, \frac{n+1}{n}\Sigma\right)
\end{aligned}
$$

Explicitly the coding distribution is

$$
\begin{aligned}
f\left(\mathbf{x}_{n+1}|\mathbf{x}^n\right) = &\sqrt{\left(\frac{n}{n+1}\right)^k} \frac{1}{\sqrt{(2\pi)^k \det\left(\Sigma\right)}} \\
&\times \exp\left\{-\frac{n}{2(n+1)}\left(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}_n\right)^T \Sigma^{-1}\left(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}_n\right)\right\}
\end{aligned} \tag{4.15}
$$

It turns out this is identical to (4.14), see Appendix A.5. The expression (4.14) is efficient for block implementation as in (4.5), while (4.15) is efficient for recursive implementation.

4.1.4.2 Vector Gaussian Case with unknown variance

Assume $\mathbf{x}_n \sim \mathcal{N}\left(\mathbf{0}, \Sigma\right)$ where the covariance matrix is unknown

$$
f\left(\mathbf{x}^n|\Sigma\right) = \frac{1}{\sqrt{(2\pi)^{kn}\det\left(\Sigma\right)^n}} \operatorname{etr}\left\{-\frac{1}{2}\hat{\Sigma}_n\Sigma^{-1}\right\}
$$

where $\hat{\mathbf{\Sigma}}_n = \sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^T$.

In order to find the MDL using SSM, notice that we can write

$$\mathbf{x}_n = \mathbf{S}\mathbf{z}_n, \quad \mathbf{z}_n \sim \mathcal{N}(0, \mathbf{I})$$

where $\mathbf{S} = \mathbf{\Sigma}^{\frac{1}{2}}$, that is $\mathbf{S}$ is *some* matrix that satisfies $\mathbf{S}\mathbf{S}^T = \mathbf{\Sigma}$. A sufficient statistic for $\mathbf{\Sigma}$ is

$$\hat{\mathbf{\Sigma}}_n = \sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^T = \mathbf{S} \sum_{i=1}^n \mathbf{z}_i \mathbf{z}_i^T \mathbf{S}^T \overset{\text{def}}{=} \mathbf{S}\mathbf{U}\mathbf{S}^T$$

Let $\hat{\mathbf{S}}_n = \hat{\mathbf{\Sigma}}_n^{\frac{1}{2}} = \mathbf{S}\mathbf{U}^{\frac{1}{2}}$. Then we can solve $\mathbf{S} = \hat{\mathbf{S}}_n \mathbf{U}^{-\frac{1}{2}}$ and $\mathbf{\Sigma} = \hat{\mathbf{S}}_n \mathbf{U}^{-1} \hat{\mathbf{S}}_n^T$. Since $\mathbf{U}^{-1}$ has Inverse-Wishart distribution $\mathbf{U}^{-1} \sim \mathcal{W}_k^{-1}(I, n)$, one can write $\Sigma \sim \mathcal{W}_k^{-1}\left(\hat{\Sigma}_n, n\right)$. Using this distribution we calculate in Appendix A.6 that

$$f\left(\mathbf{x}_{n+1} | \mathbf{x}^n\right) = \frac{1}{\pi^{\frac{k}{2}}} \frac{\det\left(\hat{\Sigma}_n\right)^{\frac{n}{2}}}{\det\left(\hat{\Sigma}_{n+1}\right)^{\frac{n+1}{2}}} \frac{\Gamma_k\left(\frac{n+1}{2}\right)}{\Gamma_k\left(\frac{n}{2}\right)} \tag{4.16}$$

where Γ_k is the multivariate gamma function [93].

On the other hand, using the normalized likelihood method we have

$$C\left(\mathbf{x}^n\right) = \frac{\Gamma_k\left(\frac{n}{2} - \frac{k+1}{2}\right)}{2^{\frac{k(k+1)}{2}} \pi^{\frac{kn}{2}} \det\left(\hat{\Sigma}_n\right)^{\frac{n}{2} - \frac{k+1}{2}}}$$

From which

$$\begin{aligned} f\left(\mathbf{x}_{n+1} | \mathbf{x}^n\right) &= \frac{C\left(\mathbf{x}^{n+1}\right)}{C\left(\mathbf{x}^n\right)} \\ &= \frac{1}{\pi^{\frac{k}{2}}} \frac{\det\left(\hat{\Sigma}_n\right)^{\frac{n}{2} - \frac{k+1}{2}}}{\det\left(\hat{\Sigma}_{n+1}\right)^{\frac{n}{2} - \frac{k}{2}}} \frac{\Gamma_k\left(\frac{n}{2} - \frac{k}{2}\right)}{\Gamma_k\left(\frac{n}{2} - \frac{k+1}{2}\right)} \end{aligned} \tag{4.17}$$

We will outline how the method can be implemented. For the coding distribution to be

well-defined, $\hat{\mathbf{\Sigma}}_n$ must have full rank, which happens when $n \geq k$ (with probability one), and the multivariate gamma function $\Gamma_k(x)$ must be defined, which happens for $x > \frac{1}{2}(k-1)$ [93]. Therefore (4.16) is well-defined for $n \geq k$ while (4.17) is defined for $n \geq 2k$, which makes the SSM very advantageous (apart from its theoretical properties). Until the coder is defined, some other coder must be used. We suggest using the scalar unknown variance coder (4.7) on each component; the first vector sample as usual needs to be encoded with the default distribution.

The coder can be used either block-based or recursively. For block-based implementation we use

$$f(\mathbf{x}^l) = \frac{1}{\pi^{\frac{k(l-k+1)}{2}}} \frac{\det\left(\hat{\Sigma}_k\right)^{\frac{k}{2}}}{\det\left(\hat{\Sigma}_{l+1}\right)^{\frac{l+1}{2}}} \frac{\Gamma_k\left(\frac{l+1}{2}\right)}{\Gamma_k\left(\frac{k}{2}\right)} f(\mathbf{x}^k).$$

For recursive implementation we use the same recursion as for linear prediction,

$$\hat{\mathbf{\Sigma}}_{n+1}^{-1} = \hat{\mathbf{\Sigma}}_n^{-1} - \frac{\hat{\mathbf{\Sigma}}_n^{-1}\mathbf{x}_{n+1}\mathbf{x}_{n+1}^T\hat{\mathbf{\Sigma}}_n^{-1}}{1 + \mathbf{x}_{n+1}^T\hat{\mathbf{\Sigma}}_n^{-1}\mathbf{x}_{n+1}}$$
$$\det\left(\hat{\mathbf{\Sigma}}_{n+1}\right) = \det\left(\hat{\mathbf{\Sigma}}_n\right)\left(1 + \mathbf{x}_{n+1}^T\hat{\mathbf{\Sigma}}_n^{-1}\mathbf{x}_{n+1}\right)$$

4.1.4.3 Vector Gaussian Case with unknown mean and variance

Assume $\mathbf{x}_n \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ where both mean and covariance matrix are unknown

$$f\left(\mathbf{x}^n|\boldsymbol{\mu}, \Sigma\right) = \frac{1}{\sqrt{(2\pi)^{kn}\det\left(\Sigma\right)^n}} \exp\left\{-\frac{1}{2}\sum_{i=1}^{n}\left(\mathbf{x}_i - \boldsymbol{\mu}\right)^T\Sigma^{-1}\left(\mathbf{x}_i - \boldsymbol{\mu}\right)\right\}$$

It is well-known [47] that sufficient statistics are $\hat{\boldsymbol{\mu}}_n = \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i$ and $\hat{\boldsymbol{\Sigma}}_n = (n-1)\,S_n = \sum_{i=1}^n (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_n)(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_n)^T$. Let $\mathbf{S}$ be a square root of $\boldsymbol{\Sigma}$, i.e., $\mathbf{S}\mathbf{S}^T = \boldsymbol{\Sigma}$. We can then write

$$\hat{\boldsymbol{\mu}}_n = \boldsymbol{\mu} + \frac{1}{\sqrt{n}}\mathbf{S}\mathbf{z}$$
$$\hat{\boldsymbol{\Sigma}}_n = \mathbf{S}\mathbf{U}\mathbf{S}^T$$

where $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, I)$ and $\mathbf{U} \sim \mathcal{W}_k(\mathbf{I}, n-1)$, $\mathbf{z}$ and $\mathbf{U}$ are independent, and $\mathcal{W}_k$ is the Wishart distribution. We solve the second equation with respect to $\mathbf{S}$ as in Section 4.1.4.2 and the first with respect to $\boldsymbol{\mu}$, to get

$$\boldsymbol{\Sigma} = \hat{\mathbf{S}}_n \mathbf{U}^{-1} \hat{\mathbf{S}}_n^T \sim \mathcal{W}_k^{-1}\left(\hat{\boldsymbol{\Sigma}}_n, n-1\right)$$
$$\boldsymbol{\mu} = \hat{\boldsymbol{\mu}}_n - \frac{1}{\sqrt{n}}\mathbf{S}\mathbf{z} = \hat{\boldsymbol{\mu}}_n - \frac{1}{\sqrt{n}}\hat{\mathbf{S}}_n \mathbf{U}^{-\frac{1}{2}}\mathbf{z} \sim \mathcal{N}\left(\hat{\boldsymbol{\mu}}_n, \frac{1}{n}\Sigma\right)$$

where $\hat{\mathbf{S}}_n$ is a square root of $\hat{\boldsymbol{\Sigma}}_n$. We can explicitly write the distributions as

$$f_{\mathbf{x}^n}(\boldsymbol{\mu}|\Sigma) = \sqrt{\frac{n^k}{(2\pi)^k \det(\Sigma)}} \exp\left\{-\frac{n}{2}(\boldsymbol{\mu} - \hat{\boldsymbol{\mu}}_n)^T \Sigma^{-1}(\boldsymbol{\mu} - \hat{\boldsymbol{\mu}}_n)\right\}$$
$$f_{\mathbf{x}^n}(\Sigma) = \frac{\det\left(\hat{\Sigma}_n\right)^{\frac{n-1}{2}}}{2^{\frac{k(n-1)}{2}}\Gamma_k\left(\frac{n-1}{2}\right)} \det(\Sigma)^{-\frac{n+k}{2}} \operatorname{etr}\left\{-\frac{1}{2}\hat{\Sigma}_n\Sigma^{-1}\right\}$$

Using these distributions, in Appendix A.7 we calculate

$$f(\mathbf{x}_{n+1}|\mathbf{x}^n) = \frac{1}{\pi^{\frac{k}{2}}}\sqrt{\left(\frac{n}{n+1}\right)^k} \frac{\det\left(\hat{\Sigma}_n\right)^{\frac{n-1}{2}}}{\det\left(\hat{\Sigma}_{n+1}\right)^{\frac{n}{2}}} \frac{\Gamma_k\left(\frac{n}{2}\right)}{\Gamma_k\left(\frac{n-1}{2}\right)}$$

and for NLM

$$f(\mathbf{x}_{n+1}|\mathbf{x}^n) = \frac{1}{\pi^{\frac{k}{2}}}\sqrt{\left(\frac{n}{n+1}\right)^k} \frac{\det\left(\hat{\Sigma}_n\right)^{\frac{n-1}{2}-\frac{k+1}{2}}}{\det\left(\hat{\Sigma}_{n+1}\right)^{\frac{n}{2}-\frac{k+1}{2}}} \frac{\Gamma_k\left(\frac{n-k-1}{2}\right)}{\Gamma_k\left(\frac{n-k-2}{2}\right)}$$

These are very similar to the case of known mean, Section 4.1.4.2. We require one more sample before the distributions become well-defined, and $\mathbf{\Sigma}_n$ is defined differently.

4.2 Asymptotic Descriptive Length for Parametrized Models

In previous section, we introduced two predictive methods (SSM and NLM) for accurate description length of parametrized models. We showed that uncertainty of model parameters increases the computational complexity of descriptive model, and this complexity is the penalty that we pay for adhering to accurate descriptive models. In this section, we would like to use asymptotic models that jointly encode the sequence and the unknown parameters, therefore computational complexity is significantly reduced.

Let $f(x^l|\boldsymbol{\theta})$ denote a pdf for the sequence x^l parametrized the k-dimensional parameter vector $\boldsymbol{\theta}$. Rissanen's famous MDL approach [43] is a way to jointly encode the sequence x^l and the unknown parameters $\boldsymbol{\theta}$. A widely known expression for codelength, frequently used in signal processing, is

$$L = -\log f(x^l|\hat{\boldsymbol{\theta}}) + \frac{k}{2}\log l \tag{4.18}$$

where $\hat{\boldsymbol{\theta}}$ is the maximum likelihood (ML) estimate. The expression (4.18) is known to be a quite good approximation for many actual MDL coding methods [81], e.g., within an $O(1)$ term under some restrictive assumptions.

One possible approach to generalizing atypicality to real-valued data is therefore to simply use the expression (4.18) as the atypical codelength. This has the advantage that we can easily take any signal processing model, count the number of unknown parameters, and then use (4.18); we believe this is a valid approach to atypicality. It also had the advantage that it is possible to derive analytical results.

As in previous chapter our main interest is to find atypical subsequences of long sequences. The main additional consideration here is that when an atypical subsequence is encoded, the decoder also needs to know the start and end of the sequence. As described in [94], the start is encoded with a special codeword of length τ bits, where $\tau \approx -\log P(\text{'atypical'})$, and the end is encoded by transmitting the length of the sequence, which using [43, 49] can be done with $\log^* l + \log c$, where c is a constant and $\log^*(l) = \log l + \log\log l + \log\log\log l + \cdots$. If we use (4.18) only the first term matters, and we get a subsequence codelength

$$L = -\log f(x^l|\hat{\boldsymbol{\theta}}) + \frac{k+2}{2}\log l + \tau \tag{4.19}$$

In principle the term τ does not matter as there are unknown constants, but τ is useful as a threshold. As shown in previous chapter the extra $\log l$ term is essential to obtain a finite atypical subsequence probability. In the following we will principally consider the subsequence problem.

4.2.1 Asymptotic MDL

In this section we assume (4.19) is used as codelength. Developing algorithms is straightforward: we just use various maximum likelihood estimators and count the number of parameters. Examples can be found in [95]. Here we will focus on performance analysis.

Consider a simple example. The typical model is a pure zero-mean Gaussian noise model with known variance σ^2. For the atypical model we let $x \sim \mathcal{N}(\mu_a, \sigma^2)$ with μ_a unknown. The typical codelength is

$$\begin{aligned} L_t(l) &= -\sum \log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{x_n^2}{2\sigma^2}\right)\right) \\ &= \frac{l}{2}\log 2\pi\sigma^2 + \sum \frac{x_n^2}{2\sigma^2 \ln 2} \end{aligned}$$

The ML estimate of the one unknown parameter μ_a is the average $\bar{x}$, and we get a codelength

using (4.19)

$$L_a(l) = \frac{l}{2}\log 2\pi\sigma^2 + \sum \frac{(x_n - \bar{x})^2}{2\sigma^2 \ln 2} + \frac{3}{2}\log l + \tau$$

The criterion for atypicality is

$$L_t(l) - L_a(l) = \frac{l\bar{x}^2}{2\sigma^2 \ln 2} - \frac{3}{2}\log l - \tau > 0$$

or

$$\left|\frac{1}{\sqrt{l}}\sum x_n\right| > \sigma\sqrt{3\ln l + (2\tau+5)\ln 2}$$

If the data is typical, $\frac{1}{\sqrt{l}}\sum x_n \sim \mathcal{N}(0,\sigma^2)$. Of key theoretical interest is the probability that a sequence generated according to the typical model is classified is atypical. One can think of this as a false alarm, but since the sequence is indistinguishable from one generated from an alternative model, we prefer the term *intrinsically atypical*.

The probability of a sequence being intrinsically atypical is upper bounded by [96]

$$\begin{aligned} P_A(l) &= 2Q\left(\sqrt{3\ln l + (2\tau+5)\ln 2}\right) \\ &\leq \exp\left(-(3\ln l + (2\tau+5)\ln 2)/2\right) \\ &= 2^{-5/2} l^{-3/2} 2^{-\tau}. \end{aligned}$$

and lower bounded by

$$\begin{aligned} P_A(l) > \frac{2}{\sqrt{2\pi(3\ln l + (2\tau+5)\ln 2)}}\left(1 - \frac{1}{3\ln l + (2\tau+5)\ln 2}\right) \\ \times \exp\left(-(3\ln l + (2\tau+5)\ln 2)/2\right) \end{aligned}$$

from which we conclude

$$\lim_{l\to\infty} \frac{\ln P_A(l)}{-\frac{3}{2}\ln l} = 1 \tag{4.20}$$

It is interesting that this is the same expression (except for constant factors) as for the

iid binary case in previous chapter. It means that, using the Gaussian mean criterion is equivalent to using the binary criterion on the *sign* of the samples. This illustrates that the discrete version of atypicality and real-valued version are part of one unified theory.

For the general vector Gaussian case, we have the following result

Theorem 11. Suppose that the typical model is $\mathcal{N}(\mathbf{s}, \mathbf{\Sigma})$ and the atypical model is $\mathcal{N}(\mathbf{s}(\boldsymbol{\theta}), \mathbf{\Sigma}(\boldsymbol{\theta}))$, where $\boldsymbol{\theta}$ is k-dimensional. Then the probability $P_A(l)$ of an intrinsically atypical subsequence is bounded by

$$\limsup_{l\to\infty} \frac{\ln P_A(l)}{-\frac{k+2}{2}\ln l} \leq 1 \tag{4.21}$$

Proof For simplicity of notation, in this proof we will assume codelength is in nats and use natural logarithms throughout. We can precode the data with the typical model, so that after precoding we can assume the typical model is $\mathcal{N}(0, \mathbf{I})$. The atypicality criterion is

$$r(\mathbf{x}) = -\ln \frac{f(\mathbf{x}|\hat{\boldsymbol{\theta}})}{f(\mathbf{x})} \geq \tau + \frac{k+2}{2}\ln l$$

The Chernoff bound now states that for any $s > 0$

$$P\left(r(\mathbf{x}) \geq \tau + \frac{k+2}{2}\ln l\right) \leq \exp(-s(\tau + \frac{k+2}{2}\ln l))M_r(s)$$

where $M_r(s) = E[e^{sr}]$. If we put $s = 1$ we obtain (4.21), provided $M_r(s)$ is bounded as $l \to \infty$. We will prove that $M_r(s) \leq K < \infty$ independent of l for any $s < 1$, which is sufficient to state (4.21) by letting $s \to 1$ sufficiently slow as $l \to \infty$.

We have

$$
\begin{aligned}
-\ln \frac{f(\mathbf{x}|\hat{\boldsymbol{\theta}})}{f(\mathbf{x})} &= \frac{1}{2}\sum_{n=1}^{l} \mathbf{x}_n^T \mathbf{x}_n \\
&\quad - \frac{1}{2}\sum_{n=1}^{l} \left(\mathbf{x}_n - \mathbf{s}(\hat{\boldsymbol{\theta}})\right)^T \hat{\boldsymbol{\Sigma}}(\boldsymbol{\theta})^{-1} \left(\mathbf{x}_n - \mathbf{s}(\hat{\boldsymbol{\theta}})\right) \\
&\quad - \frac{l}{2} \ln \det \hat{\boldsymbol{\Sigma}}(\boldsymbol{\theta})
\end{aligned}
$$

We need to upper bound this expression. Maximum likelihood estimation is given by minimizing the second and third terms over all $(\mathbf{s}(\hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\Sigma}}(\boldsymbol{\theta})), \boldsymbol{\theta} \in \mathbb{R}^k$. The set $(\mathbf{s}(\hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\Sigma}}(\boldsymbol{\theta})), \boldsymbol{\theta} \in \mathbb{R}^k$ is a manifold in $\mathbb{R}^M \times \mathbb{R}^{M^2}$. Minimizing over all (valid) vectors $\mathbb{R}^M \times \mathbb{R}^{M^2} \in \mathbb{R}^M$ can only make the term smaller, and the minimizer is of course the ML estimate, here $\mathbf{y} = (\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}})$ with $\hat{\boldsymbol{\mu}} = \frac{1}{l}\sum_{i=1}^{l} \mathbf{x}_i$, $\hat{\boldsymbol{\Sigma}} = \frac{1}{l}\sum_{i-1}^{l} (\mathbf{x}_i - \hat{\boldsymbol{\mu}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}})^T$. Thus

$$
\begin{aligned}
-\ln \frac{f(\mathbf{x}|\hat{\boldsymbol{\theta}})}{f(\mathbf{x})} &\leq \frac{1}{2}\sum_{n=1}^{l} \mathbf{x}_n^T \mathbf{x}_n \\
&\quad - \frac{1}{2}\sum_{n=1}^{l} (\mathbf{x}_n - \hat{\boldsymbol{\mu}})^T \hat{\boldsymbol{\Sigma}}^{-1} (\mathbf{x}_n - \hat{\boldsymbol{\mu}}) - \frac{l}{2} \ln \det \hat{\boldsymbol{\Sigma}} \\
&= \frac{1}{2}\sum_{n=1}^{l} \mathbf{x}_n^T \mathbf{x}_n - \frac{lM}{2} - \frac{l}{2} \ln \det \hat{\boldsymbol{\Sigma}} \\
&= \frac{l}{2}\mathrm{tr}\hat{\boldsymbol{\Sigma}} - \frac{lM}{2} - \frac{l}{2} \ln \det \hat{\boldsymbol{\Sigma}} + \frac{l}{2}\hat{\boldsymbol{\mu}}^T \hat{\boldsymbol{\mu}}
\end{aligned}
$$

and

$$
\begin{aligned}
E[e^{sr}] &\leq E\left[\exp\left(\frac{sl}{2}\mathrm{tr}\hat{\boldsymbol{\Sigma}} - \frac{slM}{2} - \frac{sl}{2}\ln\det\hat{\boldsymbol{\Sigma}}\right)\exp\left(\frac{sl}{2}\hat{\boldsymbol{\mu}}^T\hat{\boldsymbol{\mu}}\right)\right] \\
&\leq E\left[\exp\left(\frac{sl}{2}\mathrm{tr}\hat{\boldsymbol{\Sigma}} - \frac{slM}{2} - \frac{sl}{2}\ln\det\hat{\boldsymbol{\Sigma}}\right)\right] \\
&\quad \times E\left[\exp\left(\frac{sl}{2}\hat{\boldsymbol{\mu}}^T\hat{\boldsymbol{\mu}}\right)\right] \qquad (4.22)
\end{aligned}
$$

For the latter expectation we use that $\hat{\boldsymbol{\mu}} \sim \mathcal{N}(0, \frac{1}{l}\mathbf{I})$. We can therefore write

$$E\left[\exp\left(\frac{sl}{2}\hat{\boldsymbol{\mu}}^T\hat{\boldsymbol{\mu}}\right)\right] \leq \frac{\sqrt{l}}{(2\pi)^{l/2}}\int \exp\left(\frac{sl}{2}\mathbf{t}^T\mathbf{t}\right)\exp\left(-\frac{l}{2}\mathbf{t}^T\mathbf{t}\right)d\mathbf{t}$$
$$\leq K$$

for $s < 1$.

We rewrite the first expectation in (4.22) as

$$E\left[\exp\left(\frac{sl}{2}\mathrm{tr}\hat{\boldsymbol{\Sigma}} - \frac{slM}{2} - \frac{sl}{2}\ln\det\hat{\boldsymbol{\Sigma}}\right)\right]$$
$$= E\left[\exp\left(\frac{sl}{2}\mathrm{tr}\left(\frac{(l-1)}{(l-1)}\hat{\boldsymbol{\Sigma}}\right) - \frac{slM}{2} - \frac{sl}{2}\ln\det\left(\frac{(l-1)}{(l-1)}\hat{\boldsymbol{\Sigma}}\right)\right)\right]$$
$$= E\left[\exp\left(\frac{sl}{2(l-1)}\mathrm{tr}\boldsymbol{\Sigma} - \frac{slM}{2} - \frac{sl}{2}\ln\left(\frac{1}{(l-1)^M}\det\boldsymbol{\Sigma}\right)\right)\right]$$
$$= E\left[\exp\left(\frac{sl}{2(l-1)}\mathrm{tr}\boldsymbol{\Sigma} - \frac{slM}{2} + \frac{slM}{2}\ln(l-1) - \frac{sl}{2}\ln\det\boldsymbol{\Sigma}\right)\right]$$

Here $\boldsymbol{\Sigma} = (l-1)\hat{\boldsymbol{\Sigma}}$, which is known to have a Wishart distribution $\mathcal{W}_M(\mathbf{I}, l-1)$ [93] with pdf

$$f(\boldsymbol{\Sigma}) = \frac{1}{2^{(l-1)M/2}\Gamma_M(\frac{l-1}{2})}(\det\boldsymbol{\Sigma})^{(l-M-2)/2}\exp\left(-\frac{1}{2}\mathrm{tr}\boldsymbol{\Sigma}\right)$$

The expectation can now be evaluated as the integral

$$I = \alpha\int_{\boldsymbol{\Sigma}>0}\exp\left(\frac{s\frac{l}{l-1}-1}{2}\mathrm{tr}\boldsymbol{\Sigma}\right)(\det\boldsymbol{\Sigma})^{((1-s)l-M-2)/2}\,d\boldsymbol{\Sigma}$$
$$= \alpha\Gamma_M\left(\frac{(1-s)l-1}{2}\right)\left(\frac{s\frac{l}{l-1}-1}{2}\right)^{-M((1-s)l-M-2)/2-1}$$

where α is a factor independent of $\boldsymbol{\Sigma}$

$$\alpha = \frac{(l-1)^{\frac{slM}{2}}\exp\left(-\frac{slM}{2}\right)}{2^{(l-1)M/2}\Gamma_M(\frac{l-1}{2})}$$

and Γ_M is the multivariate gamma function [93]. Using Stirling's approximation repeatedly, and performing some lengthy but straightforward simplifications we then get

$$\begin{aligned} I &\sim \left(\frac{(1-s)l-1}{l-1} \right)^{M(1-M)/2} \\ &\leq K \end{aligned}$$

when $s < 1$. □

Corollary 12. Suppose that we consider a finite set of atypical signal models $\{\mathbf{s}(\boldsymbol{\theta}), \boldsymbol{\Sigma}(\boldsymbol{\theta})\}$. Then

$$\limsup_{l \to \infty} \frac{\ln P_A(l)}{-\frac{3}{2} \ln l} \leq 1$$

Proof We can use the union bound over the different models. The models with slowest decay in l will dominate for large l, and these are exactly the one-parameter models. □

On the other hand, we know from (4.20) that for the simple mean, the probability of an atypical sequence is exactly $\sim l^{-3/2}$. Thus, adding more complex models will not change this by the Corollary. This is the benefit of using MDL: searching over very complex models will not increase the probability of intrinsically atypical sequences, or in terms of anomaly detection, the false alarm probability.

4.2.2 Signal Processing Methods for Atypicality

The descriptive length of Definition 1 and Section 4.2 provides a framework for finding atypical subsequences. Into this framework one can now plug various signal processing methods. The idea is to build up a library of atypicality detectors. A main feature of atypicality is that it is not necessary to use different thresholds for different methods a lengths of atypical sequences. There is the single threshold τ which has the meaning of number of bits to encode the start of an atypical sequence, see also Theorem 3. Notice that in accordance with the strict adherence to decodability, the decoder also needs to be

informed about which atypical encoder was used. This can be done by enumerating the encoders according to increasing complexity, and using again Rissanen's universal coder for the integers [43]. This coder has the advantage that adding more elaborate models will not increase the codelength of simpler models.

In most signal processing models, the key element is the signal, i.e., the relationship between samples, rather than the exact distribution of individual samples. Often, the randomness is assumed to be Gaussian. In accordance with this, most of our methods will focus on Gaussian randomness, although this is not a requisite of atypicality. What it means is that essentially we will use only second order properties of data.

4.2.2.1 Outlier Value Detection and Uncoded transmission

One strategy for atypical coding in binary case is to simply transmit digits without coding. We would like to be able to use a similar approach for reals. This kind of approach is in particular suitable to find single values that are atypical, "outliers." A reasonable requirement is that the criterion be translation and scaling invariant. On approach would of course be to simply transmit the actual bytes representing the real numbers. However, this gives a criterion that is very dependent on the data-representation. Also, it is not very efficient: to allow overflow, the first digits should be zero. Finally, the abstract model we have adopted for the reals is a fixed number, r digits, after the period, but in principle an unlimited number of digits before the period. So, a more refined approach is needed.

At first, suppose the typical model for $x[n]$ is iid zero-mean Gaussian with σ^2 known. We then *assume* an atypical model

$$x = \theta + z, \quad z \sim \mathcal{N}(0, \sigma^2),$$

where θ is an unknown deterministic value. The reason for this is first that many real-valued data are obtained from measurements, which are usually contaminated by noise. Therefore,

the last digits in x are pure noise; the inclusion of z in the model gives a way of determining which digits are the noise-digits. Second, the inclusion of z provides a natural approach to scaling-invariance, namely by normalizing the data,

$$\tilde{x} = \sigma^{-1}x = \tilde{\theta} + \tilde{z}, \quad \mathcal{N}(0,1).$$

If all data are scaled by a factor a, $\tilde{x}$ is unchanged. In the following, assume that x is normalized, that is $\sigma^2 = 1$.

We now take a similar approach as [43]. We estimate θ – of course simply $\hat{\theta} = x$ – and then encode $\hat{\theta}$ with a precision of q fractional digits, $\hat{\theta}_q$ as an integer using Rissanen's prior for for the integers. This requires about

$$L_1 = \log^*(\lfloor 2^q|x|\rfloor + 1) + 2 + \log(c), \quad c = 2.865064$$

bits. The +2 is one bit for the sign and one to account for the fact that $\log^*$ is not integer. We then encode $x - \hat{\theta}_q$ with the optimum code for z, which requires

$$L_2 = \frac{1}{2}\log 2\pi + \sum \frac{(x - \mathrm{sign}(x)2^{-q}\lfloor 2^q|x|\rfloor)^2}{2\ln 2}$$

bits. We know want to choose q to minimize the *average* codelength over the distribution of z. The optimum value of q could conceivably depend on θ, but it turns out that one value of q works for most values of θ, see Fig. 4.4. We are primarily concerned with θ not too large. For very large values of θ, the typical code length is so large that any encoding of x would result in outlier detection. Considering this, it can be seen that $q = -1$ is a reasonable choice, although $q = 0, 1$ might also be acceptable.

To extend this to other cases than the iid zero-mean Gaussian case, we take the following approach. Based on the typical data model, we make an estimate $\hat{x}[n]$ based on $x[n-1], x[n-2], \ldots$ or $\ldots x[n-2], x[n-1], x[n+1], x[n+2] \ldots$ We then assume in the typical model that $x[n] - \hat{x}[n] \sim \mathcal{N}(0, \sigma^2)$, and use the above approach.

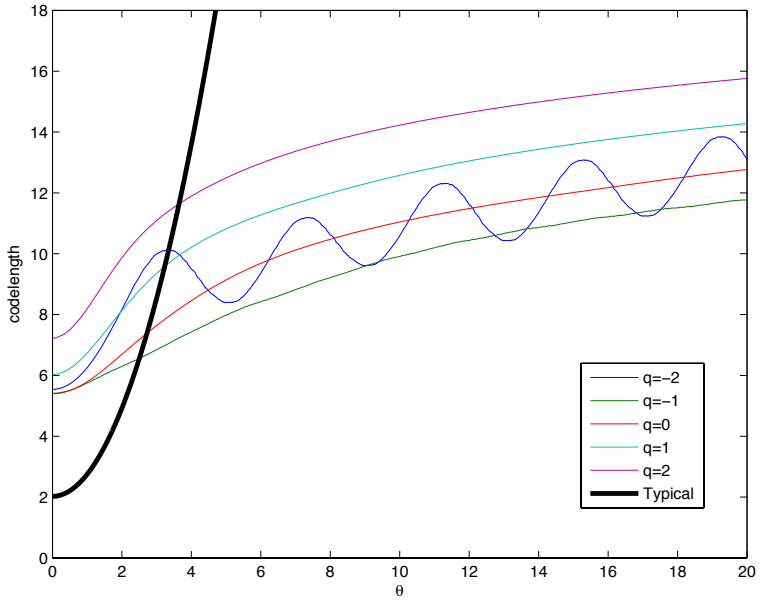

Figure 4.4: Average codelength for uncoded real encoding.

4.2.2.2 Linear Prediction

Linear prediction is a powerful method for modeling and analyzing random processes. It is used in many source coding applications, e.g., speech coding and audio coding [78]. Now, according to the Wold decomposition [71] a *stationary* random process can be decomposed into a regular and an orthogonal predictable random processes, both of which can be modeled by linear prediction. Thus, *if* the atypical segment is stationary, linear prediction should be able encode the segment well.

The application of linear prediction to atypicality is quite standard. The main consideration is computational efficiency. The codelength needs to be optimized over the prediction order, and also calculated for variable block lengths. The Levinson-Durbin algorithm [89] is well-suited for this purpose. Based on the autocorrelation sequence $\mathbf{r}$ it calculates the prediction coefficients $\mathbf{w}_m^f$ and prediction errors P_m for $m = 1, \ldots, M$, with M being the maximum prediction order. It can be verified that P_m is the actual variance of the prediction error when the *biased* autocorrelation is used, and that is all we need. In accordance with the

guiding idea of using second order properties, we assume that the prediction error is iid zero-mean Gaussian. With this we can write the codelength as

$$L_a(l) = \min_m \left\{ \frac{l}{2} \log (P_m) + \frac{l}{2} (\log 2\pi + \log e) + \frac{m+2}{2} \log l + \tau \right\}$$

4.2.2.3 Image atypicality: Sparse modeling and orthonormal bases

Assume we have the model

$$\mathbf{x} = \mathbf{f} + \mathbf{n}$$

where $\mathbf{x} \in \mathcal{R}^l$ is the noisy image (or generally, any observed data), $\mathbf{f} = \mathbf{W}\boldsymbol{\alpha}$ in which $\mathbf{W} \in \mathcal{R}^{l \times l}$ is an orthogonal matrix whose column vector are the basis elements of $\mathcal{B}$, $\boldsymbol{\alpha} \in \mathcal{R}^l$ contains the wavelet coefficients and $\mathbf{n} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$. Now for the atypical model, we suppose that the unknown signal $\mathbf{f}$ can be represented by k ($< l$) elements of the basis $\mathcal{B}$, i.e.,

$$\mathbf{f} = \mathbf{W}\boldsymbol{\alpha}^{(k)}$$

where $\boldsymbol{\alpha}^{(k)}$ is the vector of expansion coefficients of $\mathbf{f}$ with only k non-zero coefficients. We have to encode the integer k, k real-valued coefficients of $\boldsymbol{\alpha}^{(k)}$ and the value of σ^2 (total of $(k+1)$ parameters), and the deviation of the observed data $\mathbf{x}$ from the estimation $\mathbf{f} = \mathbf{W}\boldsymbol{\alpha}^{(k)}$. We also need to know the ML estimation of all the $(k+1)$ parameters. The ML estimation of the the noise variance is $\widehat{\sigma^2} = \frac{1}{l} \left\| \mathbf{x} - \mathbf{W}\boldsymbol{\alpha}^{(k)} \right\|^2$ and the ML estimation of k non-zero coefficients of $\boldsymbol{\alpha}^{(k)}$ is the largest k coefficients of $\hat{\mathbf{x}} = \mathbf{W}^T\mathbf{x}$ [97]. Now by defining the operators $\triangle^{(k)}$ and $\nabla^{(k)}$ that keep the largest magnitude k components and smallest magnitude k components respectively, and set rest to zero, we have

$$
\begin{aligned}
\widehat{\boldsymbol{\alpha}}^{(k)} &= \triangle^{(k)}(\mathbf{W}^T\mathbf{x}) \\
\widehat{\sigma^2} &= \frac{1}{l}\left\|\mathbf{x} - \mathbf{W}\boldsymbol{\alpha}^{(k)}\right\|^2 \\
&= \frac{1}{l}\left\|\hat{\mathbf{x}} - \boldsymbol{\alpha}^{(k)}\right\|^2 \\
&= \frac{1}{l}\left\|\left(\mathbf{I} - \triangle^{(k)}\right)\mathbf{W}^T\mathbf{x}\right\|^2 \\
&= \frac{1}{l}\left\|\nabla^{(l-k)}\mathbf{W}^T\mathbf{x}\right\|^2
\end{aligned}
$$

Now in addition to the value of the $k+1$ parameters that we encode using $\frac{k+1}{2}\log l$, we have to convey the index of the sequence with k non-zero elements of $\boldsymbol{\alpha}^{(k)}$ among all the sequences with k non-zero elements which requires $\log\binom{l}{k}$. Now using the upper bound from [39, (13.46)] we have $\log\binom{l}{k} \leq lH(\frac{k}{l}) + \frac{1}{2}\log l + \log\frac{\pi}{8}$, therefore

$$
\begin{aligned}
L_a = \min_{1\leq k<l}\left\{\frac{k}{2}\log l + lH(\frac{k}{l}) + \frac{l}{2}\log\left\|\nabla^{(l-k)}\mathbf{W}^T\mathbf{x}\right\|^2\right\} \\
+ 3\log l + \frac{l}{2}\log\frac{2\pi}{l} + \frac{l}{2\ln 2} + \log\frac{\pi}{8} + \tau
\end{aligned}
$$

Of course at this point, we don't know the value of k, if set it to a value close to l, then the error term $\frac{l}{2}\log\left(\left\|\nabla^{(l-k)}\mathbf{W}^T\mathbf{x}\right\|^2\right)$ will diminish but we have to pay more by encoding more value of coefficient in $\frac{5}{2}k + \frac{k}{2}\log l + lH(\frac{k}{l})$, so there is a trade off here.

Now suppose instead of one orthogonal basis $\mathcal{B}$, we have a library of orthonormal bases $\mathcal{L} = \{\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_M\}$. In fact, having a libaryr of orthogonal bases is more practical and has been considered previously, e.g., Saito in [97] consider a library of orthonormal bases in order to achieve the best denoising performance for different type of signals. For atypical encoder, the only difference in this case is that it should not only encode the number of largest coefficients (k_m) and their indexes, but also the best orthonormal base ($\mathcal{B}_m$) in the library $\mathcal{L}$. Ergo the atypical code length is

$$L_a = \min_{k_m,m} \left\{ \frac{k_m}{2} \log l + lH(\frac{k_m}{l}) + \frac{l}{2} \log \left\| \nabla^{(l-k_m)} \mathbf{W}_m^T \mathbf{x} \right\|^2 \right\}$$
$$+ 3 \log l + \frac{l}{2} \log \frac{2\pi}{l} + \frac{l}{2 \ln 2} + \log \frac{\pi}{8} + \log M + \tau$$

where the optimization is over $1 \leq k_m < l$ and $1 \leq m \leq M$.

4.3 Atypicality and Machine Learning

As we have seen in previous chapter, in many practical applications the typical model is not known, instead a long training data is provided. As a result, training data is used to learn the typical model. For discrete case, we have seen in section 3.2.3 that a universal source coder can be trained on typical data and then used as a typical encoder (training based fixed source coder). The issue with using universal source coder as a learning method is that, as they are given more training data, the dictionary (or learning tree in the case of CTW) that they generate becomes larger, therefore more memory is needed. This problem is circumvented if the learning process is done using neuron-based machine learning techniques such as neural networks, since for a fixed number of neurons, the memory requirement is independent of training data size. However, the options are limited since the desired machine learning methods should be able to achieve the codelength, therefore the probabilistic learners are of interest. Here we introduce restricted Boltzmann machine (RBM), which is a generative stochastic artificial neural network that can learn a probability distribution over a set of inputs.

4.3.1 Restricted Boltzmann Machine

A Restricted Boltzmann Machine (RBM) was first introduced as a specific type of Markov random field with two layers of neurons as binary stochastic *visible* units $\mathbf{v} \in \{0,1\}^D$ and

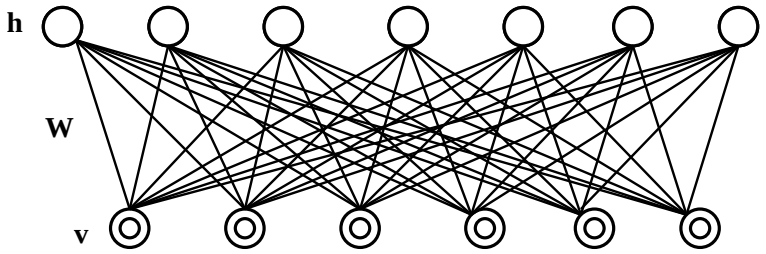

Figure 4.5: Restricted Boltzmann Machine

binary stochastic *hidden* units $\mathbf{h} \in \{0,1\}^F$ [98, 99]. Fig. 4.5 shows and example of RBM with six visible units and seven hidden units.

The energy of the state $\{\mathbf{v}, \mathbf{h}\}$ is

$$\begin{aligned} E(\mathbf{v}, \mathbf{h}; \theta) &= -\mathbf{v}^T W \mathbf{h} - \mathbf{b}^T \mathbf{v} - \mathbf{a}^T \mathbf{h} \\ &= -\sum_{i=1}^{D} \sum_{j=1}^{F} W_{ij} v_j h_j - \sum_{i=1}^{D} b_i v_i - \sum_{j=1}^{F} a_j h_j \end{aligned}$$

where $\theta = \{W, \mathbf{b}, \mathbf{a}\}$, $\mathbf{b}$ and $\mathbf{a}$ are the vector of bias terms for visible and hidden units, respectively. The joint distribution over the visible and hidden units is defined by

$$\begin{aligned} P(\mathbf{v}, \mathbf{h}; \theta) &= \frac{1}{\mathcal{Z}(\theta)} \exp\{-E(\mathbf{v}, \mathbf{h}; \theta)\} \\ \mathcal{Z}(\theta) &= \sum_{\mathbf{v}} \sum_{\mathbf{h}} \exp\{-E(\mathbf{v}, \mathbf{h}; \theta)\} \end{aligned}$$

where $\mathcal{Z}(\theta)$ is the partition function. Due to the special bipartite structure of RBM's, the hidden units can be explicitly marginalized out and the probability that the model assigns to a visible vector $\mathbf{v}$ is

$$P(\mathbf{v};\theta) = \frac{1}{\mathcal{Z}(\theta)} \sum_{\mathbf{h}} \exp\{-E(\mathbf{v},\mathbf{h};\theta)\}$$
$$= \frac{1}{\mathcal{Z}(\theta)} \exp\{\mathbf{b}^T\mathbf{v}\} \prod_{j=1}^{F} \left(1 + \exp\left\{a_j + \sum_{i=1}^{D} W_{ij} v_j\right\}\right)$$

however $\mathcal{Z}(\theta)$ cannot be expressed in a closed-form [99]. Obviously this type of learner have the potential to be used to learn the typical data (instead of training based fixed source coder of section 3.2.3). Even though the performance of a training based fixed source coder are better in redundancy sense, RBM memory requirement is independent of input size, for a fixed number of hidden units.

Gaussian RBM was then introduced as generalization of RBM to model real-valued data with hidden units [99, 100]. Consider modeling visible real-valued units $\mathbf{v} \in \mathbb{R}^D$ with binary stochastic *hidden* units $\mathbf{h} \in \{0,1\}^F$. Then the energy of the state $\{\mathbf{v},\mathbf{h}\}$ of the Gaussian RBM is defined as

$$E(\mathbf{v},\mathbf{h};\theta) = \sum_{i=1}^{D} \frac{(v_i - b_i)^2}{2\sigma_i^2} - \sum_{i=1}^{D}\sum_{j=1}^{F} W_{ij} h_j \frac{v_i}{\sigma_i} - \sum_{j=1}^{F} a_j h_j$$

where $\theta = \{W, \mathbf{b}, \mathbf{a}, \sigma^2\}$ are model parameters. The marginal distribution over the visible vector $\mathbf{v}$ is

$$P(\mathbf{v};\theta) = \frac{1}{\mathcal{Z}(\theta)} \sum_{\mathbf{h}} \exp\{-E(\mathbf{v},\mathbf{h};\theta)\}$$
$$\mathcal{Z}(\theta) = \int_{\mathbf{v}'} \sum_{\mathbf{h}} \exp\{-E(\mathbf{v},\mathbf{h};\theta)\}\, d\mathbf{v}'$$

Here we want to generalize the Gaussian RBM by considering not only real-valued visible

units but also hidden real-valued units. Suppose $\mathbf{v} \in \mathbb{R}^D$ and $\mathbf{h} \in \mathbb{R}^F$, we define the energy state $\{\mathbf{v}, \mathbf{h}\}$ of Generalized Gaussian RBM (GGRBM) as

$$\begin{aligned} E(\mathbf{v}, \mathbf{h}; \theta) &= \sum_{i=1}^{D} \frac{(v_i - b_i)^2}{2\sigma_i^2} - \sum_{i=1}^{D}\sum_{j=1}^{F} W_{ij} h_j \frac{v_i}{\sigma_i} + \sum_{j=1}^{F} \frac{(h_j - a_j)^2}{2} \\ &= \frac{1}{2}(\mathbf{v} - \mathbf{b})^T R^{-1}(\mathbf{v} - \mathbf{b}) - \mathbf{v}^T R^{-\frac{1}{2}} W \mathbf{h} + \frac{1}{2}(\mathbf{h} - \mathbf{a})^T (\mathbf{h} - \mathbf{a}) \end{aligned}$$

where $\theta = \{W, \mathbf{b}, \mathbf{a}, R\}$ and

$$R = \begin{bmatrix} \sigma_1^2 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & 0 & \cdots & 0 \\ 0 & 0 & \sigma_3^2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_D^2 \end{bmatrix}$$

The joint distribution over the visible and hidden units is defined by

$$\begin{aligned} P(\mathbf{v}, \mathbf{h}; \theta) &= \frac{1}{\mathcal{Z}(\theta)} \exp\{-E(\mathbf{v}, \mathbf{h}; \theta)\} \\ &= \frac{1}{\mathcal{Z}(\theta)} \exp\left\{-\frac{1}{2}(\mathbf{v} - \mathbf{b})^T R^{-1}(\mathbf{v} - \mathbf{b}) + \mathbf{v}^T R^{-\frac{1}{2}} W \mathbf{h} - \frac{1}{2}(\mathbf{h} - \mathbf{a})^T (\mathbf{h} - \mathbf{a})\right\} \end{aligned}$$

now using the properties of Gaussian distribution, the partition function can be find in a closed-form as follows

$$\begin{aligned} \mathcal{Z}(\theta) &= \int_{\mathbf{v}'} \int_{\mathbf{h}'} \exp\left\{-\frac{1}{2}(\mathbf{v} - \mathbf{b})^T R^{-1}(\mathbf{v} - \mathbf{b}) + \mathbf{v}^T R^{-\frac{1}{2}} W \mathbf{h} - \frac{1}{2}(\mathbf{h} - \mathbf{a})^T (\mathbf{h} - \mathbf{a})\right\} d\mathbf{h}' d\mathbf{v}' \\ &= \sqrt{\frac{\det(R)(2\pi)^{F+D}}{\det(I - WW^T)}} \\ &\times \exp\left\{\frac{1}{2}\mathbf{a}^T W^T \left(I - WW^T\right)^{-1} W\mathbf{a} + \frac{1}{2}\left(\mathbf{b}^T R^{-\frac{1}{2}} W + \mathbf{a}^T\right)\left(I - W^T W\right)^{-1} W^T R^{-\frac{1}{2}} \mathbf{b}\right\} \end{aligned}$$

using matrix inversion lemma multiple times. Hence the probability that the model assigns to a visible vector $\mathbf{v}$ is

$$\begin{aligned} P(\mathbf{v};\theta) &= \frac{\sqrt{(2\pi)^F}}{\mathcal{Z}(\theta)} \exp\left\{-\frac{1}{2}\mathbf{v}^T R^{-\frac{1}{2}}\left(I - WW^T\right)R^{-\frac{1}{2}}\mathbf{v} + \left(\mathbf{a}^T W^T R^{-\frac{1}{2}} + \mathbf{b}^T R^{-1}\right)\mathbf{v} - \frac{1}{2}\mathbf{b}^T R^{-1}\mathbf{b}\right\} \\ &= \sqrt{\frac{\det\left(I - WW^T\right)}{\det(R)(2\pi)^D}} \\ &\times \frac{\exp\left\{-\frac{1}{2}\mathbf{v}^T R^{-\frac{1}{2}}\left(I - WW^T\right)R^{-\frac{1}{2}}\mathbf{v} + \left(\mathbf{a}^T W^T R^{-\frac{1}{2}} + \mathbf{b}^T R^{-1}\right)\mathbf{v} - \frac{1}{2}\mathbf{b}^T R^{-1}\mathbf{b}\right\}}{\exp\left\{\frac{1}{2}\mathbf{a}^T W^T\left(I - WW^T\right)^{-1}W\mathbf{a} + \frac{1}{2}\left(\mathbf{b}^T R^{-\frac{1}{2}}W + \mathbf{a}^T\right)\left(I - W^T W\right)^{-1}W^T R^{-\frac{1}{2}}\mathbf{b}\right\}} \end{aligned} \tag{4.23}$$

For the introduced GGRBM, the following conditional distributions can be derived

$$P(v_i = x|\mathbf{h}) = \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left\{-\frac{\left(x - b_i - \sigma_i \sum_j h_j W_{ij}\right)^2}{2\sigma_i^2}\right\}$$

$$P(h_j = x|\mathbf{v}) = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{\left(x - a_j - \sum_i W_{ij}\frac{v_i}{\sigma_i}\right)^2}{2}\right\}$$

therefore we can write

$$
\begin{aligned}
P\left(\mathbf{v}|\mathbf{h};\theta\right) &= \prod_i P\left(v_i|\mathbf{h}\right) \\
&= \frac{1}{\sqrt{\det\left(R\right)\left(2\pi\right)^D}} \exp\left\{-\frac{1}{2}\left(\mathbf{v}-\mathbf{b}-R^{\frac{1}{2}}W\mathbf{h}\right)^T R^{-1}\left(\mathbf{v}-\mathbf{b}-R^{\frac{1}{2}}W\mathbf{h}\right)\right\} \\
P\left(\mathbf{h}|\mathbf{v};\theta\right) &= \prod_j P\left(h_j|\mathbf{v}\right) \\
&= \frac{1}{\sqrt{\left(2\pi\right)^F}} \exp\left\{-\frac{1}{2}\left(\mathbf{h}-\mathbf{a}-W^T R^{-\frac{1}{2}}\mathbf{v}\right)^T\left(\mathbf{h}-\mathbf{a}-W^T R^{-\frac{1}{2}}\mathbf{v}\right)\right\}
\end{aligned}
$$

The derivative of the log-likelihood with respect to the model parameter θ can be obtained using $P\left(\mathbf{v};\theta\right)$

$$
\begin{aligned}
\frac{\partial \log P\left(\mathbf{v};\theta\right)}{\partial W} &= \mathrm{E}_{\mathrm{P}_{\mathrm{data}}}\left[\mathrm{R}^{-\frac{1}{2}}\mathbf{v}\mathbf{h}^{\mathrm{T}}\right] - \mathrm{E}_{\mathrm{P}_{\mathrm{model}}}\left[\mathrm{R}^{-\frac{1}{2}}\mathbf{v}\mathbf{h}^{\mathrm{T}}\right] \\
\frac{\partial \log P\left(\mathbf{v};\theta\right)}{\partial \mathbf{b}} &= \mathrm{E}_{\mathrm{P}_{\mathrm{data}}}\left[\mathrm{R}^{-1}\mathbf{v}\right] - \mathrm{E}_{\mathrm{P}_{\mathrm{model}}}\left[\mathrm{R}^{-1}\mathbf{v}\right] \\
\frac{\partial \log P\left(\mathbf{v};\theta\right)}{\partial \mathbf{a}} &= \mathrm{E}_{\mathrm{P}_{\mathrm{data}}}\left[\mathbf{h}\right] - \mathrm{E}_{\mathrm{P}_{\mathrm{model}}}\left[\mathbf{h}\right]
\end{aligned}
$$

where $E_{P_{data}}\left[\cdots\right]$ is an expectation with respect to empirical data distribution and $E_{P_{model}}\left[\cdots\right]$ is an expectation with respect to model distribution. Note that learning R is constrained since all σ_i^2's must be positive. In many practical implementations, one would typically use a fixed, predetermined σ^2 [99, 100], so from now on it's assumed that σ^2 is fixed unless otherwise is explicitly mentioned. For binary RBM, since maximum likelihood estimation is intractable, leaning is done using Contrastive Divergence (CD) [99, 100]. CD can also be used here; however, the following theorem shows maximum likelihood can be calculated in a close-form for GGRBM:

Theorem 13. For an unbiased GGRBM, maximum likelihood estimation can be used to calculate the exact probability that the model assigns to a visible vector $\mathbf{v}$, i.e.

$$P(\mathbf{v}) = \sqrt{\frac{\det\left(\hat{\Delta}_{ML}\right)}{\det(R)(2\pi)^D}} \exp\left\{-\frac{1}{2}\mathbf{v}^T R^{-\frac{1}{2}} \hat{\Delta}_{ML} R^{-\frac{1}{2}} \mathbf{v}\right\}$$
$$\hat{\Delta}_{ML} = \left(\frac{1}{N}\sum_{n=1}^{N} R^{-\frac{1}{2}}\mathbf{v}[n]\,\mathbf{v}[n]^T R^{-\frac{1}{2}}\right)^{-1}$$

Proof From (4.23) the unbiased model probability is

$$P(\mathbf{v}|W) = \frac{\sqrt{(2\pi)^F}}{\mathcal{Z}(W)} \exp\left\{-\frac{1}{2}\mathbf{v}^T R^{-\frac{1}{2}}\left(I - WW^T\right) R^{-\frac{1}{2}}\mathbf{v}\right\}$$

and the log-likelihood function is

$$\ln\left[\prod_{n=1}^{N} P(\mathbf{v}[n]|W)\right] = \sum_{n=1}^{N}\left[-\frac{1}{2}\mathbf{v}[n]^T R^{-\frac{1}{2}}\left(I - WW^T\right) R^{-\frac{1}{2}}\mathbf{v}[n] - \ln\mathcal{Z}(W) + \frac{F}{2}\ln 2\pi\right]$$

and by taking derivative with respect to matrix W we have

$$\frac{\partial}{\partial W}\sum_{n=1}^{N}\left[\frac{1}{2}\mathbf{v}[n]^T R^{-\frac{1}{2}} WW^T R^{-\frac{1}{2}}\mathbf{v}[n]\right] - N\frac{\partial}{\partial W}\ln\mathcal{Z}(W) = 0$$

now by defining $d = \frac{\partial}{\partial W}$ and $A = d\mathrm{tr}\left[\mathbf{v}^{\mathrm{T}}\mathrm{R}^{-\frac{1}{2}}\mathrm{W}\mathrm{W}^{\mathrm{T}}\mathrm{R}^{-\frac{1}{2}}\mathbf{v}\right]$ we have

$$\begin{aligned}
A &= d\mathrm{tr}\left[\mathbf{v}^{\mathrm{T}}\mathrm{R}^{-\frac{1}{2}}\mathrm{W}\mathrm{W}^{\mathrm{T}}\mathrm{R}^{-\frac{1}{2}}\mathbf{v}\right] \\
&= \mathrm{tr}\left[d\left(\mathbf{v}^T R^{-\frac{1}{2}} W W^T R^{-\frac{1}{2}}\mathbf{v}\right)\right] \\
&= \mathrm{tr}\left[d\left(\mathbf{v}^T R^{-\frac{1}{2}} W\right) W^T R^{-\frac{1}{2}}\mathbf{v} + \mathbf{v}^T R^{-\frac{1}{2}} W d\left(W^T R^{-\frac{1}{2}}\mathbf{v}\right)\right] \\
&= \mathrm{tr}\left[d\left(\mathbf{v}^T R^{-\frac{1}{2}} W\right) W^T R^{-\frac{1}{2}}\mathbf{v}\right] + \mathrm{tr}\left[\mathbf{v}^T R^{-\frac{1}{2}} W d\left(W^T R^{-\frac{1}{2}}\mathbf{v}\right)\right] \\
&= \mathrm{tr}\left[\mathbf{v}^T R^{-\frac{1}{2}} d\left(W\right) W^T R^{-\frac{1}{2}}\mathbf{v}\right] + \mathrm{tr}\left[\mathbf{v}^T R^{-\frac{1}{2}} W d\left(W^T\right) R^{-\frac{1}{2}}\mathbf{v}\right] \\
&= \mathrm{tr}\left[\mathbf{v}^T R^{-\frac{1}{2}} d\left(W\right) W^T R^{-\frac{1}{2}}\mathbf{v}\right] + \mathrm{tr}\left[\left(\mathbf{v}^T R^{-\frac{1}{2}} W d\left(W^T\right) R^{-\frac{1}{2}}\mathbf{v}\right)^T\right] \\
&= \mathrm{tr}\left[\mathbf{v}^T R^{-\frac{1}{2}} d\left(W\right) W^T R^{-\frac{1}{2}}\mathbf{v}\right] + \mathrm{tr}\left[\mathbf{v}^T R^{-\frac{1}{2}} d\left(W\right) W^T R^{-\frac{1}{2}}\mathbf{v}\right] \\
&= \mathrm{tr}\left[W^T R^{-\frac{1}{2}}\mathbf{v}\mathbf{v}^T R^{-\frac{1}{2}} d\left(W\right)\right] + \mathrm{tr}\left[W^T R^{-\frac{1}{2}}\mathbf{v}\mathbf{v}^T R^{-\frac{1}{2}} d\left(W\right)\right] \\
&= 2\mathrm{tr}\left[W^T R^{-\frac{1}{2}}\mathbf{v}\mathbf{v}^T R^{-\frac{1}{2}}\right]
\end{aligned}$$

hence the first term is

$$\frac{\partial}{\partial W}\sum_{n=1}^{N}\left[\frac{1}{2}\mathbf{v}\left[n\right]^T R^{-\frac{1}{2}} W W^T R^{-\frac{1}{2}}\mathbf{v}\left[n\right]\right] = \sum_{n=1}^{N} W^T R^{-\frac{1}{2}}\mathbf{v}\left[n\right]\mathbf{v}\left[n\right]^T R^{-\frac{1}{2}}$$

in order to calculate the second term, we shall use the definition of partition function. We

have $\frac{\partial}{\partial W}\ln\mathcal{Z}(W)=\frac{\frac{\partial}{\partial W}\mathcal{Z}(W)}{\mathcal{Z}(W)}$, let $B=\frac{\partial}{\partial W}\mathcal{Z}(W)$ so

$$\begin{aligned}
B &= \frac{\partial}{\partial W}\int_{\mathbf{v}'}\int_{\mathbf{h}'}\exp\left\{-\frac{1}{2}\mathbf{v}^TR^{-1}\mathbf{v}+\mathbf{v}^TR^{-\frac{1}{2}}W\mathbf{h}-\frac{1}{2}\mathbf{h}^T\mathbf{h}\right\}d\mathbf{h}'d\mathbf{v}' \\
&= \int_{\mathbf{v}'}\int_{\mathbf{h}'}\frac{\partial}{\partial W}\exp\left\{-\frac{1}{2}\mathbf{v}^TR^{-1}\mathbf{v}+\mathbf{v}^TR^{-\frac{1}{2}}W\mathbf{h}-\frac{1}{2}\mathbf{h}^T\mathbf{h}\right\}d\mathbf{h}'d\mathbf{v}' \\
&\overset{\gamma}{=} \int_{\mathbf{v}'}\int_{\mathbf{h}'}\left(\mathbf{h}\mathbf{v}^TR^{-\frac{1}{2}}\right)\exp\left\{-\frac{1}{2}\mathbf{v}^TR^{-1}\mathbf{v}+\mathbf{v}^TR^{-\frac{1}{2}}W\mathbf{h}-\frac{1}{2}\mathbf{h}^T\mathbf{h}\right\}d\mathbf{h}'d\mathbf{v}' \\
&= \int_{\mathbf{v}'}\exp\left\{-\frac{1}{2}\mathbf{v}^TR^{-1}\mathbf{v}\right\}\left[\int_{\mathbf{h}'}\mathbf{h}\exp\left\{-\frac{1}{2}\mathbf{h}^T\mathbf{h}+\mathbf{v}^TR^{-\frac{1}{2}}W\mathbf{h}\right\}d\mathbf{h}'\right]\mathbf{v}^TR^{-\frac{1}{2}}d\mathbf{v}' \\
&= \int_{\mathbf{v}'}\exp\left\{-\frac{1}{2}\mathbf{v}^TR^{-1}\mathbf{v}\right\}\left[\sqrt{(2\pi)^F}\exp\left\{\frac{1}{2}\mathbf{v}^TR^{-\frac{1}{2}}WW^TR^{-\frac{1}{2}}\mathbf{v}\right\}W^TR^{-\frac{1}{2}}\mathbf{v}\right]\mathbf{v}^TR^{-\frac{1}{2}}d\mathbf{v}' \\
&= \sqrt{(2\pi)^F}\int_{\mathbf{v}'}\left(W^TR^{-\frac{1}{2}}\mathbf{v}\mathbf{v}^TR^{-\frac{1}{2}}\right)\exp\left\{-\frac{1}{2}\mathbf{v}^TR^{-\frac{1}{2}}\left(I-WW^T\right)R^{-\frac{1}{2}}\mathbf{v}\right\}d\mathbf{v}' \\
&= \sqrt{(2\pi)^F\det(R)}W^T\left[\int_{\mathbf{q}'}\left(\mathbf{q}\mathbf{q}^T\right)\exp\left\{-\frac{1}{2}\mathbf{q}^T\left(I-WW^T\right)\mathbf{q}\right\}d\mathbf{q}'\right] \\
&= \sqrt{\frac{(2\pi)^{F+D}\det(R)}{\det\left(I-WW^T\right)}}W^T\left(I-WW^T\right)^{-1}
\end{aligned}$$

where equation (γ) is due to $\frac{\partial}{\partial W}\left(\mathbf{v}^TR^{-\frac{1}{2}}W\mathbf{h}\right)=\frac{\partial}{\partial W}\mathrm{tr}\left(\mathbf{v}^TR^{-\frac{1}{2}}W\mathbf{h}\right)=\frac{\partial}{\partial W}\mathrm{tr}\left(\mathbf{h}\mathbf{v}^TR^{-\frac{1}{2}}W\right)=\mathbf{h}\mathbf{v}^TR^{-\frac{1}{2}}$. Hence $\frac{\partial}{\partial W}\ln\mathcal{Z}(W)=W^T\left(I-WW^T\right)^{-1}$ which proves the theorem. □

Obviously, finding maximum likelihood estimation for unsimplified models with more than one layer is intractable. Even in theorem 13 for unbiased models that the dependency of $P(\mathbf{v})$ on W is quadratic, we are able to estimate WW^T; however, estimation of W using maximum likelihood can be a NP-complete problem of multivariate quadratic equations [101, 102] of an overdetermined or underdetermined system based on values of F and D. Hence for RBM and Deep Belief Networks (DBN), we should stick to a greedy learning algorithm based on contrastive divergence algorithm [99, 100].

4.4 Experimental Results

As realistic examples, we will first consider transient detection [29] of whale vocalization using methods introduced in section 4.1, and then we have applied results of section 4.2 to find general irregularities in the stock market. But before going over experimental results, it should be mentioned how our algorithm search for atypical subsequences.

4.4.1 Algorithms

Direct implementation of atypical search requires trying to code every subsequence separately, which is prohibitively complex. Instead, we perform the search in three stages

Coarse Search. We use a tree-structured approach. At level i we divide the data into non-overlapping segments of length 2^i. These segments are then coded with the atypical coders and compared with the typical code length. This will clearly not catch all atypical sequences. Heuristically one can argue that the worst case is when an atypical sequence of length l is divided into two segments of length $\frac{l}{2}$. Each of those segments could be detected one level lower, but the threshold for length $\frac{l}{2}$ sequences is different than for length l sequences. This can be compensated for by using double length in encoder.

Fine search. Every segment that has been flagged by the coarse search is expanded, and then every subsequence is tested in an exhaustive search.

Segmentation. Once an atypical subsequence has been found, the exact beginning and end has to be found. This is done by *minimizing total code length* of the whole sequence.

4.4.2 Whale vocalization

The Station ALOHA Cabled Observatory ("ACO") is a deep sea observatory system that uses a retired telecommunications cable to transmit data collected from deep sea instruments

to shore. The ACO is located 100 km north of Oahu, Hawaii and records ocean sounds using a hydrophone from a depth of 4728 m (10 m above the seafloor) and transmits data via a fiber optic cable [103]. The data that we used were collected (with sampling freuquency of 96 kHz which was then downsampled to 8 kHz) during a proof module phase of the project conducted between February 2007 and October 2008.

The principal goal of this two years of data is whale vocalization. Fin (22 meters, up to 80 tons) and sei (12-18 meters, up to 24.6 tons) whales are known by means of visual and acoustic surveys to be present in the Hawaiian Islands during winter and spring months, but migration patterns in Hawaii are poorly understood [103]. The stereotypical fin whale vocalization is a 20 Hz downsweep with a bandwidth of 6 Hz, usually from 18-24 Hz over 1 second. This call is often produced as part of a "20 Hz doublet." Another vocalization historically attributed to the fin whales is the variable call: a 20-35 Hz, 1 second duration frequency downsweep with varying patterns, interval and frequency [103]. All these calls can have up to three echos due to reflections from sea floor and other boundaries before arriving at the receiver.

In order to assess the performance of algorithms in detecting whale calls, ground truth has been established by manual detection, which is achieved using visual inspection of spectrogram by human operator. 24 hours of manual detections for both the 20 Hz and the 20-35 Hz variable calls were recorded for each the following dates (randomly chosen): 01 March 2007, 17 November 2007, 29 May 2008, 22 August 2008, 04 September 2008 and 09 February 2008 [103]. Table 4.1 summarizes the number of fin and variable calls in each day.

	Number of Fin Calls	Number of Variable Calls
01 March 2007	9,294	1,781
17 November 2007	12,247	250
09 February 2008	15,487	2,849
29 May 2008	26	0
22 August 2008	64	36
04 September 2008	148	72

Table 4.1: Fin and variable calls in six days of manual detection

In order to analyze the performance of different detectors on such a data, first the measures Precision and Recall are defined as below

$$\text{Recall} = \frac{\text{number of correct detections}}{\text{total number of manual detections}}$$
$$\text{Precision} = \frac{\text{number of correct detections}}{\text{total number of algorithm detections}}$$

where Recall measures the probability of correctly obtained vocalizations over expected number of detections and Precision measures the probability of correctly detected vocalizations obtained by the detector. Then Precision versus Recall curve will show the detectors ability to obtain vocalizations as well as the accuracy of these detections [103].

In order to compare our atypicality method with rival approaches in transient detection, we compare its performance with Variable Threshold Page (VTP) which outperforms other similar methods in detection of non-trivial signals [104]. The VTP implementation is as a series of sequential tests according to

$$\begin{aligned} Z_n &= Z_{n-1} + g_0(x_n) - b_c \\ k &= k + 1 \\ Z_n \leq 0 &\rightarrow reset\ k = 0\ and\ Z_n = 0 \\ Z_n \geq h(k) &\rightarrow Detection \end{aligned}$$

where $g(\cdot)$ is a fixed memoryless operation without the bias term (the log-likelihood ratio), x_n is sample at time n, k refers to the number of samples since the last reset, b_c is constant bias and $h(k) = h + k(b_k - b_c)$ in which h represents the threshold in Page test, $b_k = \frac{\left(1+\frac{S_k}{k}\right)\log\left(1+\frac{S_k}{k}\right)}{\frac{S_k}{k}}$ and S_k is the total energy for a signal of length k. For atypicality method, we implement all the scalar methods of section 4.1.3. Searching for atypical sequence (in this case, whale vocalizations) has been performed in different stages (more details of algorithm implementation can be found in section 4.4.1). Fig. 4.6 shows Precision vs Recall curve for

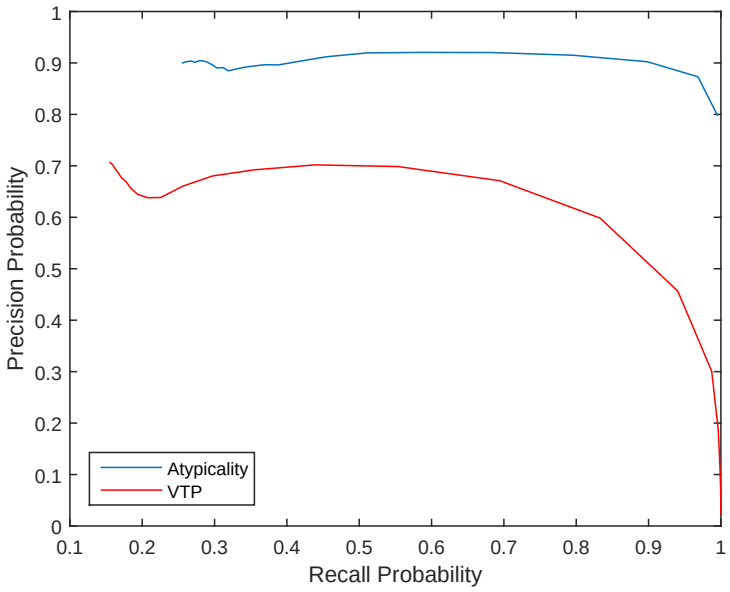

Figure 4.6: Precision vs Recall probability for all six days that manual detections are available.

both atypicality and VTP. As can be seen, atypicality outperforms VTP.

4.4.3 Stocks in S&P 500

As data we used the daily trading prices of the stocks in S&P 500 from [105] for the period 1998-2013/08/09. Data mining of the stock market has been used for fraud detection [106], but atypicality is a much more general approach that can be used to find general 'irregularities' in the stock market, the 'unknown unknowns.'

To code *typical* data we assumed that the market is efficient. The change in stock prices from day to day is therefore modeled as iid, and because we are mainly restricting ourselves to second order properties, as Gaussian iid. We took various steps to normalize the variance over time.

We ran the atypical search on these approximately 1.7 million data points. We used the differential data (day-to-day change), and applied the methods outlined section 4.2 together with the binary approach on the sign of the (differential) data. In the first run, all the

atypical data that was found was sudden big changes in stock price. These are of course indeed atypical according to our typical model. However, this kind of atypicality is hardly subtle. In the next run we therefore tried to exclude these by eliminating any jump more than 3 times standard deviation.

With $\tau = 30$ we then ended up with 106 atypical segments. Most of these turn out to still be sudden jumps in stock prices; we manually eliminated these. What we ended up with was 7 atypical segments. These are all for real estate stocks around 2000. There are 20 real estate stocks in S&P 500 [107], but 5 of these are not pure real estate. Thus, 7 out 15 pure real estate stocks behave atypically around the year 2000, see Fig. 4.7, and this is what comes out as *most* atypical. This does seem to be more than a pure coincidence.

We think we understand the reason for the indication. What seems to happen is that around 2004 the volatility of the real estate stocks increases dramatically. It seems they transition from a stable stock to a very volatile, speculative stock, and this is atypical among S&P 500 stocks. In retrospect, this seems to be an indicator of the real estate bubble.

4.4.4 Tech stocks in S&P 500

We consider a *vector* of 9 tech stocks: ADP, AMD, Apple, HP, IBM, Intel, Microsoft, Oracle and Yahoo. We process them with the methods in Section 4.2. We end up with one atypical segment in 2003, see Fig. 4.8. Now, looking at the stocks themselves, it is not clear why 2003 should be exceptional. However, looking at the NASDAQ composite index, 2003 is special in being the low point after the dot-com bubble. In some way this is reflected in the selected stock vector, although not in a simple way. Thus, atypicality has found some subtle relationship between the stocks.

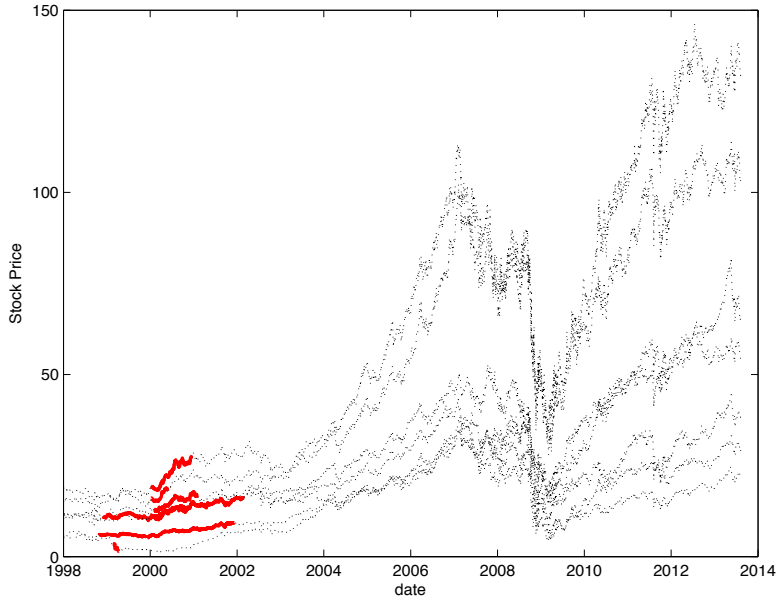

Figure 4.7: The seven real estate stocks found to be atypical. Atypical segments marked with thick red line.

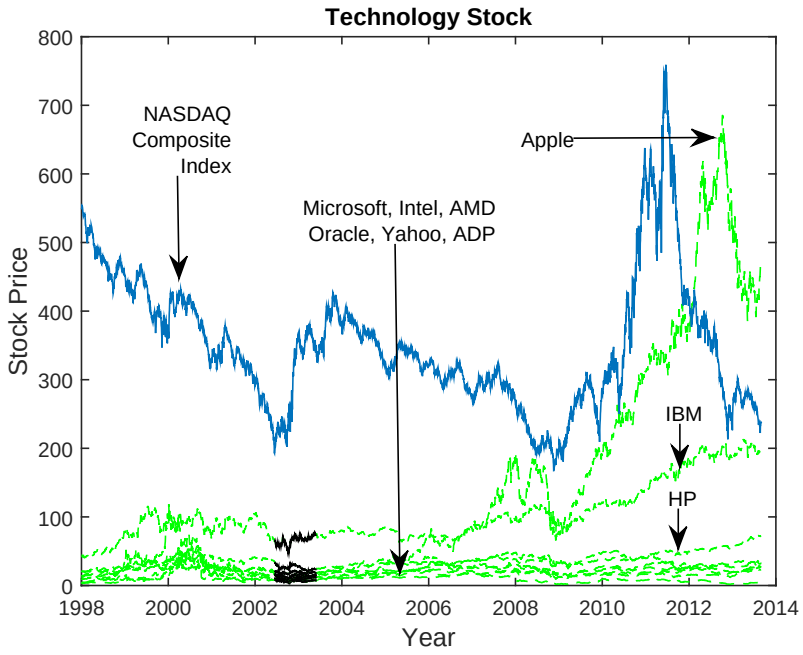

Figure 4.8: The seven real estate stocks found to be atypical. Atypical segments marked with thick red line.

5

Conclusions

In chapter 2 after defining atypicality, I outlined some anomaly detection methods previously used in the literature, but since atypicality was not fully introduced and analyzed back then, comparison with alternative approaches was postponed to this chapter. Since now we have seen atypicality framework in both discrete and real-valued cases, it's worth mentioning how atypicality can be used as tool in various detection problems as well. So in this last chapter, I'll discuss how atypicality can be deployed in other detection problems, and finally I'll sketch the future directions of atypicality.

5.1 Atypicality Application in Detection Problems

As we discussed in previous chapters, atypicality can be used in many classical detection problems such as outlier detection and anomaly detection; however unlike anomaly, atypicality is a property of data, not how the data was generated. This fact plays an important rule in underestimating atypicality and its difference with classical detection problems. We've shown in Theorem 5 that if atypicality is used as an anomaly detector, it is optimal, i.e., if the atypicality detector be given an anomalous sequence generated by a model *distinct* from the typical model, then as the length of the sequence goes toward infinity, the probability of detecting anomaly converges to 1. As a result, if the anomalous source is not *distinct* enough from the typical source, the anomalous sequence will be statistically identical to typical sequences, and there is no way to determine which model generated the sequence: the anomaly will not be detected no matter what method is used. In terms of anomaly detection, that would be called a miss. However, in terms of atypicality, this is correct: since the sequence could have been generated by the typical model, it should be classified as typical. On the other hand, if the typical model generates a sequence that is statistically different from the model, then atypicality flag will be raised for that. In anomaly detection terminology, it would be a false alarm, but based on atypicality, it would be *intrinsically* atypical. In general, atypicality is a comprehensive tool that can be used in many problems, but the aforementioned properties should be fully understood.

As mentioned in earlier chapters, due to the adherence to strict decodability at the decoder, atypicality is strongly related to minimum description length (MDL); in fact, since atypicality is based on fair comparison of typical and atypical codelength, lossless coding in MDL framework is essential. As we discussed in chapters 4, an advantage that comes with MDL cost function is its flexibility and applicability to many signal processing methods. Even though MDL plays an important rule in many detection problems, there are many approaches that use other methods. In the literature there are many detection problems that can be categorized as anomaly detection, outlier detection, transient detection, change detection

and quickest change detection. As outlined above, while what we are aiming for is not a detection problem in the traditional sense, there are still many similar applications.

Let's start with anomaly detection. Information theory and universal source coding has been used previously in anomaly detection, e.g., [1, 3–12]. These approaches haves mostly been heuristic. A more fundamental and systematic approach is information and similarity distances defined in [31]. While the similarity distance developed in [31] is not directly applicable to the atypicality setup, we can to some extent adapt it, which is useful for contrast. Suppose $K(x)$ is the Kolmogorov complexity of binary sequence x, i.e., the length of a shortest binary program x^* to compute x on an appropriate universal computer (e.g. universal Turing machine), then the conditional Kolmogorov complexity $K(x|y)$ is defined similarly as the length of a shortest program to compute sequence x if sequence y is used as an auxiliary input to the computation. Now the similarity distance between to sequence x and y is

$$d = \frac{\max\{K(y|x^*), K(x|y^*)\}}{\max\{K(x), K(y)\}}$$

Instead of being given the typical distribution, we can imagine that we are given a very long typical sequence x which is used for "training." In that case

$$d = \frac{K(x|y^*)}{K(x)} = \frac{K(x,y) - K(y)}{K(x)}$$

within a certain approximation, where $K(x,y)$ is the length of a shortest binary program that describe x and y. Suppose the typical distribution is binary iid uniform. If y is also binary iid uniform, within a constant $K(x,y) = K(x) + K(y)$, and $d = 1$. But if y is drawn from some other distribution, y cannot help describing x either, and still $d = 1$. That makes sense: two completely random sequences are not similar, whether they are from the same distribution or not. Thus, similarity distance cannot be used for anomaly detection as we have defined it: looking for 'special' sequences in the words of Kolmogorov. This is not a problem of the similarity metric; it does exactly what it is designed for, which is really deterministic similarity between sequences, appropriate for classification. Atypicality, on the other hand,

looks for statistical similarity (or rather *dissimilarity*), which seems more appropriate for anomaly detection. The reason similarity distance still gives results for anomaly detection [108] is that universal source coders approximate Kolmogorov complexity poorly!

Heuristic methods using for anomaly detection using universal source coding [1, 3–12] are mostly based on comparing codelength. Let $C(x)$ be the code length to encode the sequence x with a universal source coder. Let x be a training string and y a test sequence. We can then compare $\frac{C(x)}{|x|}$ with $\frac{C(y)}{|y|}$ or compare $C(xy)$ with $C(x)$ to detect change. The issue with this is that there are many completely dissimilar sources that have the same entropy rate. For example, if the data is binary iid and the original source has $P(X = 1) = \frac{1}{3}$ while the new source has $P(X = 1) = \frac{2}{3}$. In that case, the optimum code for the original source and the optimum code for the new source have the same length. On the other hand, atypicality will immediately distinguish such sequences. Could there be cases where this approach detects an anomaly, but atypicality does not? The following Proposition essentially says that this is not the case:

Proposition 2. Let $\mathcal{S}_1$ be an ergodic source with entropy rate $H(\mathcal{S}_1)$, and let $L_1(\mathcal{S}_1)$ be the length of the optimum source code for $\mathcal{S}_1$. We know that $L_1(\mathcal{S}_1) = H(\mathcal{S}_1)$. Let $\mathcal{S}_2$ be another ergodic source with optimum source code length L_2, and suppose $|L_1(\mathcal{S}_1) - L_1(\mathcal{S}_2)| > 0$. Then $L_1(\mathcal{S}_2) - L_2(\mathcal{S}_2) > 0$.

Proof We have

$$
\begin{aligned}
L_1(\mathcal{S}_1) &= \lim_{n\to\infty} \sum_{x_1,\ldots x_n} -p_1(x_1,\ldots,x_n) \log p_1(x_1,\ldots,x_n) \\
L_1(\mathcal{S}_2) &= \lim_{n\to\infty} \sum_{x_1,\ldots x_n} -p_2(x_1,\ldots,x_n) \log p_1(x_1,\ldots,x_n) \\
L_2(\mathcal{S}_2) &= \lim_{n\to\infty} \sum_{x_1,\ldots x_n} -p_2(x_1,\ldots,x_n) \log p_2(x_1,\ldots,x_n)
\end{aligned}
$$

Fix n, and suppose

$$\begin{aligned}
&\sum_{x_1,\ldots x_n} -p_1(x_1,\ldots,x_n)\log p_1(x_1,\ldots,x_n) \\
&\quad - \sum_{x_1,\ldots x_n} -p_2(x_1,\ldots,x_n)\log p_1(x_1,\ldots,x_n) \\
=&\sum_{x_1,\ldots x_n} (p_2(x_1,\ldots,x_n)-p_1(x_1,\ldots,x_n))\log p_1(x_1,\ldots,x_n) \\
\neq&0
\end{aligned}$$

Then $p_1(x_1,\ldots,x_n) - p_2(x_1,\ldots,x_n) \neq 0$ for some $x_1,\ldots x_n$. But then

$$\begin{aligned}
&\sum_{x_1,\ldots x_n} -p_2(x_1,\ldots,x_n)\log p_1(x_1,\ldots,x_n) \\
&\quad - \sum_{x_1,\ldots x_n} -p_2(x_1,\ldots,x_n)\log p_2(x_1,\ldots,x_n) \\
=&D(p_2(x_1,\ldots x_n)\|p_1(x_1,\ldots x_n)) \\
>&0
\end{aligned}$$

□

by [39]. This can then be extended to the limit.

In the setting of the proposition, $|L_1(\mathcal{S}_1) - L_2(\mathcal{S}_2)|$ corresponds to comparing codelength, while $L_1(\mathcal{S}_2) - L_2(\mathcal{S}_2)$ corresponds to atypicality. Notice that we can clearly not conclude the opposite direction. We could have $L_1(\mathcal{S}_2) - L_2(\mathcal{S}_2) > 0$ while $|L_1(\mathcal{S}_1) - L_1(\mathcal{S}_2)| = 0$. Thus the atypicality approach is never weaker than detecting change in codelength.

Another type of detection problem is transient detection [29, 104, 109, 110]. In many signal processing application, it is of interest to detect short-duration statistical changes in observed data. For a parametric class of probability distribution $\{f(x|\theta) : \theta \in \Theta\}$ and for an unknown

n_s and n_d the following two hypotheses are considered

$$H_0 : x_1^l \sim f\left(x|\theta_0\right)$$
$$H_1 : x_1^{n_s-1} \sim f\left(x|\theta_0\right),\ x_{n_s}^{n_{d-1}} \sim f\left(x|\theta_1\right),\ x_{n_d}^l \sim f\left(x|\theta_0\right)$$

If θ_0 and θ_1 are known, the Page test is optimal for this in the sense that by using a generalized likelihood ratio test (GLRT); given an average wait between false alarms, it minimizes the worst-case average delay to detection [104]. However in many applications, there is either no information about θ_1 or it varies from one transient signal to another. In this case, it is shown that Variable Threshold Page (VTP) gives a reliable result [104, 109]. There are also other approaches of transient detection based on Nuttall's power-law detector that are often used in the literature [109, 110]. In general atypicality will outperform this methods since it not only allows more comprehensive class of models, but also it can take advantage of various powerful signal processing methods such as filterbanks and linear prediction to find transient signals with various statistics.

Another famous detection problem is change detection [111–113]. In this problem, the goal is to detect the point when the nature of the data stream significantly changes. This problem is well-studied in the case that the class of probability distributions is known, but the parameter value for both before and after change are unknown. This problem is related to *concept drift* topic [114] for both abrupt and incremental changes. For a parametric class of probability distribution $\{f\left(x|\theta\right) : \theta \in \Theta\}$ and for a given t, the following two hypotheses are considered

$$H_0 : x_1^l \sim f\left(x|\theta_0\right)$$
$$H_1 : x_1^t \sim f\left(x|\theta_1\right),\ x_{t+1}^l \sim f\left(x|\theta_2\right)$$

where θ_0, θ_1 and θ_2 are not known in advance and the goal is to find the change point t, if it exists. In [112] this problem is solved by using an MDL-change statistic based on the

total codelength difference of the data with change and without change. In this approach normalized maximum likelihood (NML) encoder is used to calculate the codelength of data. Also similar approach is used in [111] to detect clustering structural changes. In another work [113], in order to find multiple change point, an MDL cost function is minimized over possible numbers and locations of change points, and using a proposed pruning algorithm to remove unlikely change points, computational cost reduced to linear in the number of observations. Atypicality can also be applied for this type of problems, even though in essence it would look for change points (start point of atypical subsequence) with the assumption of $\theta_0 = \theta_1 \neq \theta_2$.

In quickest change detection (QCD) problem, the goal is to find a change points in which the statistical properties of a stochastic process undergo a change. Even though at first glance this problem looks very similar to atypicality, it is completely different problem setup. Basically for a given false alarm probability, QCD algorithms aim to find the change point that minimizes average detection delay. Based on the available information regarding the distribution of change point, QCD methods are divided into Bayesian quickest change detection algorithm (BQCD) algorithms and minimax quickest change detection (MQCD) algorithms. In BQCD it is assumed that the distribution of the change point is not known, on the other hand in MQCD the change point is modeled as deterministic but unknown positive integer. In essence, MQCD is closely related to Page-based transient detection methods and can be used interchangeably. A good survey can be found in [115]. In atypicality perspective, as it was alluded earlier, there is no such a thing as false alarm in traditional sense; however, if atypicality is going to be used as a quickest change detection method, there will be a trade-off between false positive rate and delay with respect to atypicality algorithm threshold τ. In another word, the smaller the algorithm threshold, the smaller the detection delay and the larger the false positive rate.

Finally, another related problem (that was studied mostly more discrete case) is universal outlier detection [116, 117]. Even though in these works they considered different levels of knowledge about typical and outlier distributions and also various number of outliers (exactly one, at most one, more than one and unknown), the most comprehensive setup is for the

scenario that no knowledge about both typical and outlier models are available. Based on the problem setup, generalized likelihood test or its modification is shown to be universally exponentially consistent (except for the case where the outlier sequences can be distinctly distributed with their total number being unknown). This problem is closely related to our unsupervised atypicality framework, which is one of the topics that should be further developed.

5.2 Future Directions

The principal direction in future of atypicality is in the learning process of typical model, briefly introduced in section 4.3. As it was alluded, the main challenge in atypicality is lack of prior knowledge about typical model in practice, even though in the theory of atypicality, typical model (and therefore optimum encoder of typical data) is assumed to be known. In many practical experiments, instead of typical model, a training data is available. In section 3.2.3 we showed how a universal source coder can be developed to learn the typical model using training data and then applied as a training based fixed source coder. Then in section 4.3 we mentioned that while a training based fixed source coder has outstanding performance in the redundancy sense, it requires more memory as it is given more data. So the probabilistic learners such as restricted Boltzmann machine (RBM) that can learn a probability distribution (and hence calculate codelength) over a set of inputs are of interest. The memory requirement of these neuron based learners are independent of input data size for a fixed number of neurons. Further analysis and experiment of atypicality criterion when machine learning is used to learn the typical model is a promising future direction, especially for real-valued data.

Next step would be improving and expanding of unsupervised case, in which even training data is not available, instead we are given a collection of data, of which some might be atypical. We then need to separate it into typical and atypical data. This problem is stated in chapter 3.5 and a heuristic algorithm is proposed; however, this framework should be

developed profoundly.

A further generalization of unsupervised case is scenario that we are just given (say) a real-valued time-series that does not follow any parametric model, and we would like to find rare *unknown* atypical subsequence. The uncertainty in this general setup makes it hard to even formulate this problem, but it can be considered as an ultimate direction of atypicality research.

Bibliography

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection for discrete sequences: A survey," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 24, no. 5, pp. 823 –839, may 2012.

[2] D. Rumsfeld, *Known and Unknown: A Memoir*. Penguin, 2011.

[3] S. Evans, B. Barnett, S. Bush, and G. Saulnier, "Minimum description length principles for detection and classification of ftp exploits," in *Military Communications Conference, 2004. MILCOM 2004. 2004 IEEE*, vol. 1, oct.-3 nov. 2004, pp. 473 – 479 Vol. 1.

[4] N. Wang, J. Han, and J. Fang, "An anomaly detection algorithm based on lossless compression," in *Networking, Architecture and Storage (NAS), 2012 IEEE 7th International Conference on*, 2012, pp. 31–38.

[5] W. Lee and D. Xiang, "Information-theoretic measures for anomaly detection," in *Security and Privacy, 2001. S P 2001. Proceedings. 2001 IEEE Symposium on*, 2001, pp. 130–143.

[6] I. Paschalidis and G. Smaragdakis, "Spatio-temporal network anomaly detection by assessing deviations of empirical measures," *Networking, IEEE/ACM Transactions on*, vol. 17, no. 3, pp. 685–697, 2009.

[7] C.-K. Han and H.-K. Choi, "Effective discovery of attacks using entropy of packet dynamics," *Network, IEEE*, vol. 23, no. 5, pp. 4–12, 2009.

[8] P. Baliga and T. Lin, "Kolmogorov complexity based automata modeling for intrusion detection," in *Granular Computing, 2005 IEEE International Conference on*, vol. 2, 2005, pp. 387–392 Vol. 2.

[9] H. Shahriar and M. Zulkernine, "Information-theoretic detection of sql injection attacks," in *High-Assurance Systems Engineering (HASE), 2012 IEEE 14th International Symposium on*, 2012, pp. 40–47.

[10] Y. Xiang, K. Li, and W. Zhou, "Low-rate ddos attacks detection and traceback by using new information metrics," *Information Forensics and Security, IEEE Transactions on*, vol. 6, no. 2, pp. 426–437, 2011.

[11] F. Pan and W. Wang, "Anomaly detection based-on the regularity of normal behaviors," in *Systems and Control in Aerospace and Astronautics, 2006. ISSCAA 2006. 1st International Symposium on*, 2006, pp. 6 pp.–1046.

[12] E. Eiland and L. Liebrock, "An application of information theory to intrusion detection," in *Information Assurance, 2006. IWIA 2006. Fourth IEEE International Workshop on*, 2006, pp. 16 pp.–134.

[13] M. Thottan and C. Ji, "Anomaly detection in ip networks," *Signal Processing, IEEE Transactions on*, vol. 51, no. 8, pp. 2191 – 2204, aug. 2003.

[14] ——, "Proactive anomaly detection using distributed intelligent agents," *Network, IEEE*, vol. 12, no. 5, pp. 21 –27, sep/oct 1998.

[15] S. Singh, H. Tu, W. Donat, K. Pattipati, and P. Willett, "Anomaly detection via feature-aided tracking and hidden markov models," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 39, no. 1, pp. 144 –159, jan. 2009.

[16] C. H. Bennett, P. Gacs, M. Li, P. M. B. Vitanyi, and W. H. Zurek, "Information distance," *IEEE Transactions on Information Theory*, vol. 44, no. 4, pp. 1407–1423, July 1998.

[17] C. Bennett, P. Gacs, M. Li, P. M. Vitanyi, and W. Zurek, "Information distance," *Information Theory, IEEE Transactions on*, vol. 44, no. 4, pp. 1407 –1423, july 1998.

[18] S. Budalakoti, A. Srivastava, and M. Otey, "Anomaly detection and diagnosis algorithms for discrete symbol sequences with applications to airline safety," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 39, no. 1, pp. 101–113, Jan 2009.

[19] J. Unnikrishnan, V. Veeravalli, and S. Meyn, "Minimax robust quickest change detection," *Information Theory, IEEE Transactions on*, vol. 57, no. 3, pp. 1604 –1614, march 2011.

[20] V. Raghavan and V. Veeravalli, "Quickest change detection of a markov process across a sensor array," *Information Theory, IEEE Transactions on*, vol. 56, no. 4, pp. 1961 –1981, april 2010.

[21] V. Veeravalli, "Decentralized quickest change detection," *Information Theory, IEEE Transactions on*, vol. 47, no. 4, pp. 1657 –1665, may 2001.

[22] O. Hadjiliadis, H. Zhang, and H. Poor, "One shot schemes for decentralized quickest change detection," *Information Theory, IEEE Transactions on*, vol. 55, no. 7, pp. 3346 –3359, july 2009.

[23] Y. Mei, "Information bounds and quickest change detection in decentralized decision systems," *Information Theory, IEEE Transactions on*, vol. 51, no. 7, pp. 2669 –2681, july 2005.

[24] S. Blostein, "Quickest detection of a time-varying change in distribution," *Information Theory, IEEE Transactions on*, vol. 37, no. 4, pp. 1116 –1122, jul 1991.

[25] Y. Liu and S. Blostein, "Quickest detection of an abrupt change in a random sequence with finite change-time," *Information Theory, IEEE Transactions on*, vol. 40, no. 6, pp. 1985 –1993, nov 1994.

[26] E. Rafajandowicz, M. Pawlak, and A. Steland, "Nonparametric sequential change-point detection by a vertically trimmed box method," *Information Theory, IEEE Transactions on*, vol. 56, no. 7, pp. 3621 –3634, july 2010.

[27] U. Niesen and A. Tchamkerten, "Tracking stopping times through noisy observations," *Information Theory, IEEE Transactions on*, vol. 55, no. 1, pp. 422 –432, jan. 2009.

[28] B. Chen and P. Willett, "Detection of hidden markov model transient signals," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 36, no. 4, pp. 1253 – 1268, oct 2000.

[29] C. Han, P. Willett, B. Chen, and D. Abraham, "A detection optimal min-max test for transient signals," *Information Theory, IEEE Transactions on*, vol. 44, no. 2, pp. 866 –869, mar 1998.

[30] A. Tartakovsky and M. Pollak, "Nearly minimax changepoint detection procedures," in *Information Theory Proceedings (ISIT), 2011 IEEE International Symposium on*, 31 2011-aug. 5 2011, pp. 2676 –2680.

[31] M. Li, X. Chen, X. Li, B. Ma, and P. Vitanyi, "The similarity metric," *Information Theory, IEEE Transactions on*, vol. 50, no. 12, pp. 3250 – 3264, dec. 2004.

[32] P. Vitanyi, "Information distance in multiples," *Information Theory, IEEE Transactions on*, vol. 57, no. 4, pp. 2451 –2456, april 2011.

[33] R. Meo, "Maximum independence and mutual information," *Information Theory, IEEE Transactions on*, vol. 48, no. 1, pp. 318 –324, jan 2002.

[34] M. Malik, "Heart rate variability," *Annals of Noninvasive Electrocardiology*, vol. 1, no. 2, pp. 151–181, April 1996.

[35] M. F. Hilton, R. A. Bates, K. R. Godfrey, and et al., "Evaluation of frequency and time-frequency spectral analysis of heart rate variability as a diagnostic marker of the sleep apnea syndrome," *Med. Biol. Eng. Comput.*, vol. 37, no. 6, pp. 760–769, November 1999.

[36] N. V. Thakor and Y. S. Zhu, "Application of adaptive filtering to ECG analysis: Noise cancellation and arrhythmia detection," *IEEE Trans. on Biomed. Eng.*, vol. 38, pp. 785–794, August 1991.

[37] J. F. Thayer, S. S. Yamamoto, and J. F. Brosschot, "The relationship of autonomic imbalance, heart rate variability and cardiovascular disease risk factors," *International Journal of Cardiology*, vol. 141, no. 2, pp. 122–131, May 2009.

[38] J. Fondon and H. Garner, "Probing human cardiovascular congenital disease using transgenic mouse models," *Proc Natl Acad Sci U S A*, vol. 101, no. 52, pp. 18 058–63, 2004.

[39] T. Cover and J. Thomas, *Information Theory, 2nd Edition.* John Wiley, 2006.

[40] M. Li and P. Vitányi, *An Introduction to Kolmogorov Complexity and Its Applications*, 3rd ed. Springer, 2008.

[41] A. Nies, *Computability and Randomness.* Oxford University Press, 2009.

[42] J. Rissanen, "Complexity of strings in the class of markov sources," *Information Theory, IEEE Transactions on*, vol. 32, no. 4, pp. 526 – 532, jul 1986.

[43] ——, "A universal prior for integers and estimation by minimum description length," *The Annals of Statistics*, no. 2, pp. 416–431, 1983.

[44] ——, "Universal coding, information, prediction, and estimation," *Information Theory, IEEE Transactions on*, vol. 30, no. 4, pp. 629 – 636, jul 1984.

[45] ——, "Stochastic complexity and modeling," *The Annals of Statistics*, no. 3, pp. 1080–1100, Sep. 1986.

[46] S. M. Kay, *Fundamentals of Statistical Signal Processing, Volume I: Estimation Theory*. Prentice-Hall, 1993.

[47] L. L. Scharf, *Statistical Signal Processing: Detection, Estimation, and Time Series Analysis*. Addison-Wesley, 1990.

[48] G. Shamir, "On the mdl principle for i.i.d. sources with large alphabets," *Information Theory, IEEE Transactions on*, vol. 52, no. 5, pp. 1939–1955, May 2006.

[49] P. Elias, "Universal codeword sets and representations of the integers," *Information Theory, IEEE Transactions on*, vol. 21, no. 2, pp. 194 – 203, mar 1975.

[50] H. V. Poor, *An Introduction to Signal Detection and Estimation*. Springer-Verlag, 1994.

[51] A. Dembo and O. Zeitouni, *Large Deviations Techniques and Applications*. Springer, 1998.

[52] E. L. Lehmann, *Testing Statistical Hypotheses*. Springer, 2005.

[53] N. Merhav, "On the minimum description length principle for sources with piecewise constant parameters," *Information Theory, IEEE Transactions on*, vol. 39, no. 6, pp. 1962 –1967, nov 1993.

[54] J. Rissanen, "Modeling by shortest data description," *Automatica*, pp. 465–471, 1978.

[55] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *Information Theory, IEEE Transactions on*, vol. 23, no. 3, pp. 337 – 343, may 1977.

[56] ——, "Compression of individual sequences via variable-rate coding," *Information Theory, IEEE Transactions on*, vol. 24, no. 5, pp. 530 – 536, sep 1978.

[57] M. Effros, K. Visweswariah, S. Kulkarni, and S. Verdu, "Universal lossless source coding with the burrows wheeler transform," *Information Theory, IEEE Transactions on*, vol. 48, no. 5, pp. 1061 –1081, may 2002.

[58] J. Cleary and I. Witten, "Data compression using adaptive coding and partial string matching," *Communications, IEEE Transactions on*, vol. 32, no. 4, pp. 396 – 402, apr 1984.

[59] A. Moffat, "Implementing the ppm data compression scheme," *Communications, IEEE Transactions on*, vol. 38, no. 11, pp. 1917 –1921, nov 1990.

[60] M. Titchener, "Deterministic computation of complexity, information and entropy," in *Information Theory, 1998. Proceedings. 1998 IEEE International Symposium on*, aug 1998, p. 326.

[61] K. Hamano and H. Yamamoto, "A randomness test based on t-codes," in *Information Theory and Its Applications, 2008. ISITA 2008. International Symposium on*, dec. 2008, pp. 1 –6.

[62] K. Kawaharada, K. Ohzeki, and U. Speidel, "Information and entropy measurements on video sequences," in *Information, Communications and Signal Processing, 2005 Fifth International Conference on*, 0-0 2005, pp. 1150 –1154.

[63] U. Speidel, "A note on the estimation of string complexity for short strings," in *Information, Communications and Signal Processing, 2009. ICICS 2009. 7th International Conference on*, dec. 2009, pp. 1 –5.

[64] U. Speidel, R. Eimann, and N. Brownlee, "Detecting network events via t-entropy," in *Information, Communications Signal Processing, 2007 6th International Conference on*, dec. 2007, pp. 1 –5.

[65] U. Speidel and T. Gulliver, "An analytic upper bound on t-complexity," in *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*, july 2012, pp. 2706 –2710.

[66] J. Yang and U. Speidel, "String parsing-based similarity detection," in *Information Theory Workshop, 2005 IEEE*, aug.-1 sept. 2005, p. 5 pp.

[67] P. A. J. Volf and F. M. J. Willems, "Switching between two universal source coding algorithms," in *Data Compression Conference, 1998. DCC '98. Proceedings*, 1998, pp. 491–500.

[68] P. Jacquet and W. Szpankowski, "Limiting distribution of lempel ziv'78 redundancy," in *Information Theory Proceedings (ISIT), 2011 IEEE International Symposium on*, 31 2011-aug. 5 2011, pp. 1509 –1513.

[69] F. M. J. Willems, Y. Shtarkov, and T. Tjalkens, "The context-tree weighting method: basic properties," *Information Theory, IEEE Transactions on*, vol. 41, no. 3, pp. 653–664, 1995.

[70] F. Willems, "The context-tree weighting method: extensions," *Information Theory, IEEE Transactions on*, vol. 44, no. 2, pp. 792–798, Mar 1998.

[71] G. R. Grimmett and D. R. Stirzaker, *Probability and Random Processes, Third Edition*. Oxford University Press, 2001.

[72] W. Liu, I. Park, and J. Principe, "An information theoretic approach of designing sparse kernel adaptive filters," *Neural Networks, IEEE Transactions on*, vol. 20, no. 12, pp. 1950–1961, Dec 2009.

[73] "Department of Statistics, UC Berkeley," http://www.stat.berkeley.edu/.

[74] "PhysioBank ATM," http://physionet.org/cgi-bin/atm/ATM/.

[75] R. J. Britten, "DNA sequence insertion and evolutionary variation in gene regulation," *Proceedings of the National Academy of Sciences*, vol. 93, no. 18, pp. 9374–9377, 1996. [Online]. Available: http://www.pnas.org/content/93/18/9374.abstract

[76] J. Z. K. Khattak, S. Rauf, Z. Anwar, H. M. Wahedi, and T. Jamil, "Recent advances in genetic engineering - a review," *Current Research Journal of Biological Sciences*, vol. 4, no. 1, 2012.

[77] A. Høst-Madsen, E. Sabeti, and C. Walton, "Information theory for atypical sequences," in *IEEE Information Theory Workshop (ITW'13), Seville, Spain*, 2013.

[78] F. Ghido and I. Tabus, "Sparse modeling for lossless audio compression," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 21, no. 1, pp. 14–28, Jan 2013.

[79] P. D. Grünwald, *The minimum description length principle*. MIT press, 2007.

[80] A. Barron, J. Rissanen, and B. Yu, "The minimum description length principle in coding and modeling," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2743–2760, 1998.

[81] P. D. Gr unwald, *The Minimum Description Length Principle*. MIT Press, 2007.

[82] W. Kotlowski and P. Grunwald, "Sequential normalized maximum likelihood in log-loss prediction," in *Information Theory Workshop (ITW), 2012 IEEE*. IEEE, 2012, pp. 547–551.

[83] T. Roos and J. Rissanen, "On sequentially normalized maximum likelihood models," 2008.

[84] J. Rissanen, *Information and complexity in statistical modeling*. Springer Science & Business Media, 2007.

[85] R. J. Larsen and M. L. Marx, *An introduction to mathematical statistics and its applications*, 1986.

[86] J. Rissanen, *Stochastic complexity in statistical inquiry*. World scientific, 1998, vol. 15.

[87] H. Scheffe, "A useful convergence theorem for probability distributions," *Ann. Math. Statist.*, vol. 18, no. 3, pp. 434–438, 09 1947. [Online]. Available: http://dx.doi.org/10.1214/aoms/1177730390

[88] G. Forchini, "The density of the sufficient statistics for a gaussian ar(1) model in terms of generalized functions," *Statistics & Probability Letters*, vol. 50, no. 3, pp.

237 – 243, 2000. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167715200001115

[89] S. Haykin, *Adaptive Filter Theory*. Prentice-Hall, 2001.

[90] H. V. Henderson and S. R. Searle, "On deriving the inverse of a sum of matrices," *Siam Review*, vol. 23, no. 1, pp. 53–60, 1981.

[91] S. K. Mitra and Y. Kuo, *Digital signal processing: a computer-based approach*. McGraw-Hill New York, 2006, vol. 2.

[92] S. Mallat, *A wavelet tour of signal processing: the sparse way*. Academic press, 2008.

[93] R. J. Muirhead, *Aspects of multivariate statistical theory*. John Wiley & Sons, 2009, vol. 197.

[94] A. Høst-Madsen, E. Sabeti, and C. Walton, "Data discovery and anomaly detection using atypicality," *IEEE Transactions on Information Theory*, submitted, available at http://tinyurl.com/jndmvo7.

[95] A. Høst-Madsen and E. Sabeti, "Atypical information theory for real-valued data," in *Proceedings of International Symposium on Information Theory*, 2015.

[96] S. Verdú, *Multiuser Detection*. Cambridge, UK: Cambridge University Press, 1998.

[97] N. Saito, "Simultaneous noise suppression and signal compression using a library of orthonormal bases and the minimum-description-length criterion," in *SPIE's International Symposium on Optical Engineering and Photonics in Aerospace Sensing*. International Society for Optics and Photonics, 1994, pp. 224–235.

[98] P. Smolensky, "Information processing in dynamical systems: Foundations of harmony theory," DTIC Document, Tech. Rep., 1986.

[99] R. Salakhutdinov, "Learning deep generative models," Ph.D. dissertation, University of Toronto, 2009.

[100] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[101] E. Thomae and C. Wolf, "Solving underdetermined systems of multivariate quadratic equations revisited," in *International Workshop on Public Key Cryptography*. Springer, 2012, pp. 156–171.

[102] N. Courtois, A. Klimov, J. Patarin, and A. Shamir, "Efficient algorithms for solving overdefined systems of multivariate polynomial equations," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2000, pp. 392–407.

[103] K. Silver, "A passive acoustic automated detector for sei and fin whale calls," Master's thesis, University of Hawaii, 2014.

[104] Z. J. Wang and P. Willett, "A variable threshold page procedure for detection of transient signals," *IEEE transactions on signal processing*, vol. 53, no. 11, pp. 4397–4402, 2005.

[105] QuantQuote, "Free Historical Stock Data," https://quantquote.com/historical-stock-data.

[106] K. Golmohammadi and O. Zaiane, "Data mining applications for fraud detection in securities market," in *Intelligence and Security Informatics Conference (EISIC), 2012 European*, 2012, pp. 107–114.

[107] REIT, "REITs in S&P Indexes," http://www.reit.com/investing/investing-tools/reits-sp-indexes.

[108] E. Keogh, S. Lonardi, and C. A. Ratanamahatana, "Towards parameter-free data mining," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2004, pp. 206–215.

[109] Z. Wang and P. Willett, "A performance study of some transient detectors," *Signal*

Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on], vol. 48, no. 9, pp. 2682–2685, Sep. 2000.

[110] Z. Wang and P. K. Willett, "All-purpose and plug-in power-law detectors for transient signals," *Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on]*, vol. 49, no. 11, pp. 2454–2466, Nov. 2001.

[111] S. Hirai and K. Yamanishi, "Detecting changes of clustering structures using normalized maximum likelihood coding," in *Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 2012, pp. 343–351.

[112] K. Yamanishi and K. Miyaguchi, "Detecting gradual changes from data stream using mdl-change statistics," in *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 2016, pp. 156–163.

[113] R. Killick, P. Fearnhead, and I. A. Eckley, "Optimal detection of changepoints with a linear computational cost," *Journal of the American Statistical Association*, vol. 107, no. 500, pp. 1590–1598, 2012.

[114] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, p. 44, 2014.

[115] V. V. Veeravalli and T. Banerjee, "Quickest change detection," *Academic press library in signal processing: Array and statistical signal processing*, vol. 3, pp. 209–256, 2013.

[116] Y. Li, S. Nitinawarat, and V. V. Veeravalli, "Universal outlier hypothesis testing," *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 4066–4082, 2014.

[117] ——, "Universal outlier detection," in *Information Theory and Applications Workshop (ITA), 2013*. IEEE, 2013, pp. 1–5.

Appendix

A.1 Proof of Theorem 3

Without loss of generality we can assume $n = 0$. For some $l_0 > 0$ let $\mathcal{I}_{l_0}$ be the set of subsequences containing X_0 of length $l \leq l_0$. For $i \in \mathcal{I}_{l_0}$ let $l(i)$ be the length of the subinterval. From Theorem 1 we know that $P_A(i) \leq 2^{-\tau+1}\frac{1}{l^{3/2}}K(l,\tau)$ and therefore

$$\sum_{i\in\mathcal{I}_{l_0}} P_A(i) \leq K2^{-\tau}$$

for some constant K. This argument does not work if we allow arbitrarily long subsequences, because the sum is divergent. However, we can write

$$P_A(X_0) \leq \sum_{i\in\mathcal{I}_{l_0}} P_A(i) \leq K2^{-\tau} + P_{A,l_0}(X_0)$$

where $P_{A,l_0}(X_0)$ is the probability that X_0 is in an atypical subsequence of at least length l_0. The proof will be to bound $P_{A,l_0}(X_0)$.

Define the following events

$$\overline{A}(n_1, l) = \left\{ \sum_{i=n_1}^{n_1+l-1} X_i - p > \sqrt{lpq \ln 2(2\tau + 3\log l)} \right\}$$
$$\underline{A}(n_1, l) = \left\{ \sum_{i=n_1}^{n_1+l-1} X_i - p < -\sqrt{lpq \ln 2(2\tau + 3\log l)} \right\}$$

For $p = \frac{1}{2}$ we can rewrite

$$\begin{aligned} &\sum_{i=n_1}^{n_1+l-1} X_i - p > \sqrt{lpq \ln 2(2\tau + 3\log l)} \\ &= \sum_{i=n_1}^{n_1+l-1} (2X_i - 1) > \sqrt{l \ln 2(2\tau + 3\log l)} \end{aligned} \tag{A.1.1}$$

For ease of notation define

$$\upsilon(l) = \left\lceil \sqrt{l \ln 2(2\tau + 3\log l)} \right\rceil$$

Then using the union bound we can write

$$\begin{aligned}
P_A(X_0) \leq & \sum_{n_1=-\infty}^{-1} P\left(\bigcup_{l=-n_1+1}^{\infty} \overline{A}(n_1,l)\right) \\
& + \sum_{n_1=-\infty}^{-1} P\left(\bigcup_{l=-n_1+1}^{\infty} \underline{A}(n_1,l)\right) \\
\leq & 2 \sum_{n_1=-\infty}^{-1} P\left(\bigcup_{l=-n_1+1}^{\infty} \overline{A}(n_1,l)\right) \\
= & 2 \sum_{n_1=-\infty}^{-1} 1 - P\left(\bigcap_{l=-n_1+1}^{\infty} \overline{A}^c(n_1,l)\right) \\
= & 2 \sum_{n_1=-\infty}^{-1} \left(1 - \right. \\
& \left. \prod_{l=-n_1+1}^{\infty} P\left(\overline{A}^c(n_1,l) \middle| \bigcap_{\ell=n-n_1+1}^{l-1} \overline{A}^c(n_1,\ell)\right)\right)
\end{aligned}$$

where we have excluded the length one sequence consisting of X_0 itself. Now consider $P\left(\overline{A}^c(n_1,l) \left| \bigcap_{\ell=-n_1+1}^{l-1} \overline{A}^c(n_1,\ell)\right.\right) = 1 - P\left(\overline{A}(n_1,l) \left| \bigcap_{\ell=-n_1+1}^{l-1} \overline{A}^c(n_1,\ell)\right.\right)$.

We can put this problem in the framework of simple random walks [71], and we will use this to upper bound the probability $P\left(\overline{A}(n_1,l) \left| \bigcap_{\ell=-n_1+1}^{l-1} \overline{A}^c(n_1,\ell)\right.\right) = 1 - P\left(\overline{A}^c(n_1,l) \left| \bigcap_{\ell=-n_1+1}^{l-1} \overline{A}^c(n_1,\ell)\right.\right)$. Let $S_l = \sum_{i=n_1}^{n_1+l-1}(2X_i - 1)$. This probability can be interpreted as the probability that the random walk passes $\upsilon(l)$ given that it was below $\upsilon(\ell)$ at times $-n_1 < \ell < l$. But since the random walk can increase by at most one, and since the threshold is increasing with l, that means that at time l we must have $S_l = \upsilon(l)$. Furthermore, it is easy to see that the probability is upper bounded by the probability that $S_l = \upsilon(l)$ given that the random walk

is below $\upsilon(l)$ at times $-n_1 < \ell < l$. Thus

$$\begin{aligned}
&P\left(\overline{A}(n_1,l)\,\middle|\,\bigcap_{\ell=-n_1+1}^{l-1}\overline{A}^c(n_1,\ell)\right)\\
&\leq P\left(S_l=\upsilon(l)\,|S_\ell<\upsilon(l),-n_1<\ell<l\,\right)\\
&=\frac{P\left(S_l=\upsilon(l),S_\ell<\upsilon(l),-n_1<\ell<l\right)}{P\left(S_\ell<\upsilon(l),-n_1<\ell<l\right)}\\
&\leq\frac{P\left(S_l=\upsilon(l),S_\ell<\upsilon(l),-n_1<\ell<l\right)}{P\left(S_\ell<\upsilon(l),0\leq\ell<l\right)}
\end{aligned}\tag{A.1.2}$$

The denominator can be interpreted as the probability that the maximum of the random walk stays below $\upsilon(l)$, which by Theorem 1 can be expressed by

$$\begin{aligned}
P_D(l)&=P\left(S_\ell<\upsilon(l),0\leq\ell<l\right)\\
&=1-2P(S_{l-1}\geq\upsilon(l+1)-1)-P(S_{l-1}=\upsilon(l)-1)\\
&\geq 1-2^{-\tau+c}l^{-3/2}\\
&\geq\frac{1}{2}
\end{aligned}$$

for τ and l sufficiently large, and where c is some constant. Since, as discussed at the start of the proof, we can assume that $l \geq l_0$, we can choose l_0 large enough that this is satisfied; furthermore, since $P_N(l)$ is increasing in τ, we can choose l_0 independent of τ as long as τ is sufficiently large.

We will next upper bound the numerator in (A.1.2). This is the probability that we have a path that has stayed below $\upsilon(l)$ at steps $-n_1 < \ell < l$, but then at step l hits $\upsilon(l)$. We will count such paths. We divide them into two groups that we count separately. The first group are all paths that start at zero and hit $\upsilon(l)$ *first time* after l steps. The second group is more easily described in reverse time. Those are paths that start at $\upsilon(l)$ at step l, then stay below $\upsilon(l)$ until time $\tilde{n} < 0$, when they hit $\upsilon(l)$ again, and finally hit 0 at time n_1. According to

[71, Section 3.10] we can count these paths by

$$N = \frac{\upsilon(l)}{l} N_l(0, \upsilon(l)) + \sum_{t=\upsilon(l)}^{-n_1+1} \frac{1}{l-t-1} N_{l-t-1}(1,0) N_t(0, \upsilon(l)) \tag{A.1.3}$$

Where $N_n(a,b)$ are the number of length n paths between a and b.

We need to upper bound the probability $P(S_n = k)$ that a path starting a 0 hits k after n steps. We use [71, Section 3.10] and [39, 13.2] to get

$$\begin{aligned}
P\left(S_n = k\right) =& N_n(0,k) 2^{-n} \\
=& \begin{pmatrix} n \\ \frac{1}{2}(n+k) \end{pmatrix} 2^{-n} \\
\leq& \sqrt{\frac{n}{\pi \frac{1}{4}(n+k)(n-k)}} 2^{nH\left(\frac{\frac{1}{2}(n+k)}{n}\right)} 2^{-n} \\
=& \sqrt{\frac{4n}{\pi(n^2-k^2)}} 2^{nH\left(\frac{\frac{1}{2}(n+k)}{n}\right)} 2^{-n}
\end{aligned}$$

We can bound the power of the exponent to 2 as follows

$$\begin{aligned}
& nH\left(\frac{\frac{1}{2}(n+k)}{n}\right) - n \\
=& n\left(H\left(\frac{1}{2} + \frac{1}{2}\frac{k}{n}\right) - 1\right) \\
\leq& -\frac{2}{\ln 2} n \left(\frac{1}{2}\frac{k}{n}\right)^2 \\
=& -\frac{1}{2\ln 2}\frac{k^2}{n}
\end{aligned} \tag{A.1.4}$$

Thus,

$$P\left(S_n = k\right) \leq \frac{2}{\sqrt{\pi n}} e_2\left(-\frac{1}{2\ln 2}\frac{k^2}{n}\right)$$

where $e_2(x) = 2^x$.

We will use this to bound the probability of the later set of paths in (A.1.3). We can bound

$$\begin{aligned}
P_2(n_1, l) &= \sum_{t=\upsilon(l)}^{-n_1+1} \frac{1}{l-t-1} N_{l-t-1}(1,0) N_t(0, \upsilon(l)) 2^{-(l-1)} \\
&\leq \sum_{t=\upsilon(l)}^{-n_1+1} \frac{1}{l-t-1} \frac{2}{\sqrt{\pi(l-t-1)}} \\
&\quad \times e_2\left(-\frac{1}{2\ln 2}\frac{1}{l-t-1}\right) N_t(0, \upsilon(l)) 2^{-t} \\
&\leq \frac{4}{\pi(l+n_1-2)^{3/2}} \sum_{t=\upsilon(l)}^{-n_1+1} P(S_t = \upsilon(l))
\end{aligned}$$

Here the sum $\sum_{t=\upsilon(l)}^{-n_1+1} P(S_t = \upsilon(l))$ when looked at in reverse time can be interpreted as the probability of a path starting at $\upsilon(l)$ hits zero before time $-n_1+1$. We can the write this as (See [71, Section 3.10])

$$\begin{aligned}
\sum_{t=\upsilon(l)}^{-n_1+1} P(S_t = \upsilon(l)) &= P(M_{-n_1+1} \geq \upsilon(l)) \\
&\leq 2P(S_{-n_1+1} \geq \upsilon(l))
\end{aligned}$$

We can use the proof of Theorem 1, specifically (3.12) to bound this by

$$P(M_{-n_1+1} \geq \upsilon(l)) \leq \exp\left(-\frac{2\upsilon(l)^2}{-n_1+1}\right)$$

Then

$$P_2(n_1, l) \leq \frac{K}{(l+n_1-2)^{3/2}} \exp\left(-\frac{\upsilon(l)^2}{2(-n_1+1)}\right) \sqrt{-n_1+1}$$

We will next bound the probability of the paths in the first term in (A.1.3). We have

$$\begin{aligned}
P\left(S_l = \upsilon(l)\right) \leq & \sqrt{\frac{4l}{\pi(l^2 - \upsilon(l)^2)}} e_2\left(-\frac{1}{2\ln 2}\frac{\upsilon(l)^2}{l}\right) \\
& \sqrt{\frac{4}{\pi l\left(1 - \frac{\upsilon(l)^2}{l^2}\right)}} e_2\left(-\frac{1}{2\ln 2}\frac{\upsilon(l)^2}{l}\right) \\
= & \frac{2}{\sqrt{\pi\left(1 - \frac{\upsilon(l)^2}{l^2}\right)}} 2^{-\tau} l^{-2}
\end{aligned}$$

and

$$\begin{aligned}
P_1(l) = & \frac{\upsilon(l)}{l} P\left(S_l = \upsilon(l)\right) \\
\leq & \sqrt{\frac{\ln 2(2\tau + 3\log l)}{l}} \frac{2}{\sqrt{\pi\left(1 - \frac{\upsilon(l)^2}{l^2}\right)}} 2^{-\tau} l^{-2} \\
\leq & \sqrt{\frac{8\tau \ln 2}{\pi\left(1 - \frac{\upsilon(l)^2}{l^2}\right)}} 2^{-\tau} l^{-5/2} \\
& + \sqrt{\frac{12 \ln l}{\pi\left(1 - \frac{\upsilon(l)^2}{l^2}\right)}} \sqrt{\frac{4}{\pi}} 2^{-\tau} l^{-5/2}
\end{aligned}$$

Thus

$$\begin{aligned}
& \ln\left(\prod_{l=-n_1+1}^{\infty} P\left(\overline{A}^c(n_1, l) \middle| \bigcap_{\ell=-n_1+1}^{l-1} \overline{A}^c(n_1, \ell)\right)\right) \\
\geq & \sum_{l=-n_1+1}^{\infty} \ln\left(1 - \frac{P_1(l) - P_2(n_1, l)}{P_D(l)}\right) \\
\geq & K \sum_{l=-n_1+1}^{\infty} -P_1(l) - P_2(n_1, l) \\
\doteq & S(-n_1, \tau)
\end{aligned}$$

and

$$\begin{aligned} P_A(X_0) \leq & 2 \sum_{n_1=-\infty}^{-1} 1 - e^{S(-n_1,\tau)} \\ \leq & 2K \sum_{n_1=-\infty}^{-1} \sum_{l=-n_1+1}^{\infty} P_1(l) \\ & + 2K \sum_{n_1=-\infty}^{-1} \sum_{l=-n_1+1}^{\infty} P_2(n_1, l) \end{aligned} \tag{A.1.5}$$

where $K > 0$ is some constant.

First we evaluate the sum of $P_1(l)$. The term $\frac{v(l)^2}{l^2}$ is decreasing in l, so for sufficiently large l_1, $\frac{v(l)^2}{l^2} \leq \frac{1}{2}$. We can evaluate the sum separately for $l_0 \leq l \leq l_1$ and for $l > l_1$. Convergence depends only the latter tail. The threshold l_1 is increasing with τ. If for example we put $l_1 = 8\tau \ln 2$, i.e., proportional to τ, we have $\frac{v(l)^2}{l^2} \leq \frac{1}{2}$ for $\tau > 10$. Therefore

$$\sum_{l=l_0}^{l_1} P_1(l) \leq K\tau 2^{-\tau}$$

For $l > l_1$ we can write

$$P_1(l) = \sqrt{\frac{16\tau \ln 2}{\pi}} 2^{-\tau} l^{-5/2} + \sqrt{\frac{24 \ln l}{\pi}} \sqrt{\frac{4}{\pi}} 2^{-\tau} l^{-5/2}$$

Then (for $-n_1 + 1 > l_1$)

$$\begin{aligned} & K \sum_{l=-n_1+1}^{\infty} P_1(l) \\ \leq & k_1 2^{-\tau} \sqrt{\frac{\ln(n - n_1)}{(-n_1)^3}} \\ & + k_2 2^{-\tau} \mathrm{erfc}\left(\sqrt{\frac{3}{2} \ln(-n_1)}\right) \\ & + k_3 2^{-\tau} \sqrt{\frac{\tau}{(-n_1)^3}} \end{aligned} \tag{A.1.6}$$

where $k_i > 0$ are some constants and where we have used

$$\begin{aligned}
\sum_{l=k}^{\infty} l^{-5/2} &\leq \int_{k-1}^{\infty} x^{-5/2} dx = \frac{2}{3(k-1)^{3/2}} \\
\sum_{l=k}^{\infty} \sqrt{\ln l} l^{-5/2} &\leq \int_{k-1}^{\infty} \sqrt{\ln x} x^{-5/2} dx \\
&= \frac{1}{9} \left(\sqrt{6\pi} \mathrm{erfc} \left[\sqrt{\frac{3}{2} \ln k - 1} \right] + \frac{6\sqrt{\ln k - 1}}{(k-1)^{3/2}} \right)
\end{aligned}$$

as it can be verified that all three sums, when (A.1.6) is inserted in (A.1.5), are convergent, using $\sum_{k=1}^{\infty} f(k) \leq f(1) + \int_1^{\infty} f(x) dx$.

We bound the second sum in (A.1.5),

$$\begin{aligned}
&\sum_{n_1=-\infty}^{-1} \sum_{l=-n_1+1}^{\infty} P_2(n_1, l) \\
&= \sum_{n_1=-\infty}^{-1} \sum_{l=-n_1+1}^{\infty} \frac{8\sqrt{-n_1+1}}{\pi(l+n_1-2)^{3/2}} \exp\left(-\frac{v(l)^2}{2(-n_1+1)} \right)
\end{aligned}$$

We can ignore the small constants and write

$$\begin{aligned}
P &= \sum_{n_1=-\infty}^{-1} \sum_{l=-n_1+1}^{\infty} \frac{8}{\pi(l+n_1)^{3/2}} \exp\left(-\frac{\upsilon(l)^2}{2(-n_1)}\right) \sqrt{-n_1} \\
&= \sum_{n_1=-\infty}^{-1} \sum_{l=-n_1+1}^{\infty} \frac{8\sqrt{-n_1}}{\pi(l+n_1)^{3/2}} 2^{-\frac{l\tau}{-n_1}} l^{-\frac{3l}{-n_1}} \\
&\leq \int_1^{\infty} \int_t^{\infty} \frac{8\sqrt{t}}{\pi(l-t)^{3/2}} 2^{-\frac{l\tau}{t}} l^{-\frac{3l}{t}} dl dt \\
&= \int_1^{\infty} \int_1^{\infty} \frac{8\sqrt{t}}{\pi t^{3/2}(\tilde{l}-1)^{3/2}} 2^{-\tilde{l}\tau} \tilde{l}^{-3\tilde{l}} t^{-3\tilde{l}} t d\tilde{l} dt \\
&= \int_1^{\infty} \int_1^{\infty} \frac{8}{\pi(\tilde{l}-1)^{3/2}} 2^{-\tilde{l}\tau} \tilde{l}^{-3\tilde{l}} t^{-3\tilde{l}} d\tilde{l} dt \\
&= \int_1^{\infty} \left(\int_1^{\infty} t^{-3\tilde{l}} dt\right) \frac{8}{\pi(\tilde{l}-1)^{3/2}} 2^{-\tilde{l}\tau} \tilde{l}^{-3\tilde{l}} d\tilde{l} \\
&= \int_1^{\infty} \frac{1}{3\tilde{l}-1} \frac{8}{\pi(\tilde{l}-1)^{3/2}} 2^{-\tilde{l}\tau} \tilde{l}^{-3\tilde{l}} d\tilde{l} \\
&= 2^{-\tau} \int_1^{\infty} \frac{1}{3\tilde{l}-1} \frac{8}{\pi(\tilde{l}-1)^{3/2}} 2^{-(\tilde{l}-1)\tau} \tilde{l}^{-3\tilde{l}} d\tilde{l}
\end{aligned}$$

The remaining integral is clearly convergent, and decreasing in τ. Therefore $P \leq K2^{-\tau}$

A.2 Proof of Proposition 1

We can assume that $n = 0$. We will continue with the random walk framework from the proof of Theorem 3. Define the event

$$\bar{A}_l = \left\{ \sum_{i=-l+1}^{0} (2X_i - 1) > \sqrt{l \ln 2(2\tau + \alpha \log l)} \right\}$$
$$\underline{\mathrm{A}}_l = \left\{ \sum_{i=-l+1}^{0} (2X_i - 1) < \sqrt{l \ln 2(2\tau + \alpha \log l)} \right\}$$

and

$$\upsilon(l) = \left\lceil \sqrt{l \ln 2(2\tau + \alpha \log l)} \right\rceil$$

Then

$$P_A(X_0) \geq P\left(\bigcup_{l=0}^{\infty} \bar{A}_l \cup \bigcup_{l=0}^{\infty} \underline{\mathrm{A}}_l \right)$$

Namely, we declare that X_0 is atypical if it is the endpoint of an atypical sequence $\{x[-l], x[-l+1], \ldots, x[0]\}$ for some l. Clearly, X_0 could be the start or midpoint of an atypical sequence, so this a rather loose lower bound. Now we can write

$$\begin{aligned}
&P\left(\bigcup_{l=0}^{\infty} \bar{A}_l \cup \bigcup_{l=0}^{\infty} \underline{\mathrm{A}}_l \right) \\
&= 1 - P\left(\bigcap_{l=0}^{\infty} \bar{A}_l^c \cap \bigcap_{l=0}^{\infty} \underline{\mathrm{A}}_l^c \right) \\
&= 1 - \prod_{l=0}^{\infty} P\left(\bar{A}_l^c \cap \underline{\mathrm{A}}_l^c \,\middle|\, \bigcap_{k=0}^{l-1} \bar{A}_k^c \cap \bigcap_{k=0}^{l-1} \underline{\mathrm{A}}_k^c \right) \\
&= 1 - \prod_{l=0}^{\infty} \left[1 - P\left(\bar{A}_l \cup \underline{\mathrm{A}}_l \,\middle|\, \bigcap_{k=0}^{l-1} \bar{A}_k^c \cap \bigcap_{k=0}^{l-1} \underline{\mathrm{A}}_k^c \right) \right]
\end{aligned}$$

Consider the probability $P\left(\bar{A}_l \cup \underline{\mathrm{A}}_l \,\middle|\, \bigcap_{k=0}^{l-1} \bar{A}_k^c \cap \bigcap_{k=0}^{l-1} \underline{\mathrm{A}}_k^c \right)$. The only way the conditional event can happen is if $S_{l-1} = \upsilon(l) - 1$ and $X_l = 1$ or $S_{l-1} = -\upsilon(l) + 1$ and $X_l = -1$. Here

we have

$$
\begin{aligned}
&P\left(S_{l-1}=\upsilon(l)-1\right)\\
&=N_{l-1}(0,\upsilon(l)-1)2^{-l+1}\\
&=\begin{pmatrix} l-1 \\ \frac{1}{2}(l+\upsilon(l)-2) \end{pmatrix}2^{-l+1}\\
&\geq\sqrt{\frac{l-1}{2(l+\upsilon(l)-2)(l-\upsilon(l))}}2^{(l-1)H\left(\frac{\frac{1}{2}(l+\upsilon(l)-2)}{l-1}\right)}2^{-l+1}\\
&=\sqrt{\frac{l-1}{2((l-1)^2-(\upsilon(l)-1)^2)}}2^{(l-1)H\left(\frac{\frac{1}{2}(l+\upsilon(l)-2)}{l-1}\right)}2^{-l+1}\\
&\geq\sqrt{\frac{1}{2l}}2^{(l-1)H\left(\frac{\frac{1}{2}(l+\upsilon(l)-2)}{l-1}\right)}2^{-l+1}
\end{aligned}
$$

Here

$$
\begin{aligned}
&(l-1)H\left(\frac{\frac{1}{2}(l+\upsilon(l)-2)}{l-1}\right)-l+1\\
=&(l-1)\left(H\left(\frac{1}{2}+\frac{1}{2}\frac{\upsilon(l)-1}{l-1}\right)-1\right)\\
\geq&-\frac{2}{\ln 2}(l-1)\left(\frac{1}{2}\frac{\upsilon(l)-1}{l-1}\right)^2+(l-1)o\left(\left(\frac{\upsilon(l)-1}{l-1}\right)^3\right)\\
=&-\frac{1}{2\ln 2}\frac{(\upsilon(l)-1)^2}{l-1}+\frac{(\upsilon(l)-1)^3}{(l-1)^2}\epsilon\left(\frac{1}{l}\right)\\
\geq&-\frac{1}{2\ln 2}\left(\frac{\upsilon(l)^2}{l}+\frac{\upsilon(l)^2}{(l-1)^2}+\frac{1}{l-1}\right)+\frac{\upsilon(l)^3}{l^2}\epsilon\left(\frac{1}{l}\right)\\
=&-\frac{1}{2\ln 2}\frac{\upsilon(l)^2}{l}+\frac{\upsilon(l)^3}{l^2}\epsilon\left(\frac{1}{l}\right)\\
\geq&-\frac{1}{2\ln 2}\frac{\upsilon(l)^2}{l}-\frac{\upsilon(l)^3}{l^2}
\end{aligned}
$$

Where the last inequality is only true for l sufficiently large, as for some l_0 we have $\forall l > l_0 : |\epsilon(l^{-1})| < 1$. Then

$$\begin{aligned} P\left(S_{l-1} = v(l) - 1\right) &\geq \sqrt{\frac{1}{2l}} 2^{-\tau} l^{-\frac{\alpha}{2}} 2^{-\frac{v(l)^3}{l^2}} \\ &= \sqrt{\frac{1}{2}} 2^{-\tau} l^{-\frac{\alpha+1}{2}} 2^{-\frac{v(l)^3}{l^2}} \end{aligned}$$

And

$$\begin{aligned} &\ln\left(1 - P\left(\bigcup_{l=0}^{\infty} \bar{A}_l \cup \bigcup_{l=0}^{\infty} \underline{\mathrm{A}}_l\right)\right) \\ \leq &\sum_{l=1}^{\infty} \ln\left(1 - \sqrt{\frac{1}{2}} 2^{-\tau} l^{-\frac{\alpha+1}{2}} 2^{-\frac{v(l)^3}{l^2}}\right) \\ \leq &\sum_{l=0}^{\infty} -\sqrt{\frac{1}{2}} 2^{-\tau} l^{-\frac{\alpha+1}{2}} 2^{-\frac{v(l)^3}{l^2}} \end{aligned}$$

Here $\lim_{l\to\infty} \frac{v(l)^3}{l^2} = 0$. So, for example, for l sufficiently large, $\frac{v(l)^3}{l^2} \leq 1$. Then

$$\begin{aligned} &\ln\left(1 - P\left(\bigcup_{l=0}^{\infty} \bar{A}_l \cup \bigcup_{l=0}^{\infty} \underline{\mathrm{A}}_l\right)\right) \\ \leq &\sum_{l=l_0}^{\infty} -\frac{1}{2}\sqrt{\frac{1}{2}} 2^{-\tau} l^{-\frac{\alpha+1}{2}} \end{aligned}$$

This is divergent for $\alpha \leq 1$ proving that $P\left(\bigcup_{l=0}^{\infty} \bar{A}_l \cup \bigcup_{l=0}^{\infty} \underline{\mathrm{A}}_l\right) = 1$.

A.3 Proof of Theorem 4

Since we consider all state machines with the number of states up to a certain maximum, this must also include the state machine with a single state. This is equivalent to the iid model in Section 3.1, and we therefore get the lower bound in (3.25). The proof will be to upper bound the probability. As in Section 3.1 we use $\log l + \tau$ bits to indicate beginning and end of atypical sequences. The probability that a sequence x^l is atypical therefore is

$$\begin{aligned} P_A(l) &= P(I(x^l) + \log l + \tau > l) \\ &= P(\bigcup_{f_j} -\log P(x^l|f_j) + \log^* j + c + \log l + \tau > l) \\ &\leq \bigcup_{f_j} P(-\log P(x^l|f_j) + \log l + \tau > l) \end{aligned}$$

We will prove that $P(-\log P(x^l|f_j) + \log l + \tau > l) \leq K_j l^{-(k+2)/2}$ for constants K_j and k the number of states in the state machine, and since the slowest decay dominates, we get the upper bound for (3.25).

For a fixed state f the code length according to [42, (3.6)] is

$$L(x^l|f) = \sum_s \log \begin{pmatrix} n_s(x^l) \\ n_{0|s}(x^l) \end{pmatrix} + \sum_s \log(n_s(x^l) + 1)$$

where $n_s(x^l)$ denotes the number of occurrences of state s in x^l and $n_{0|s}(x^l)$ the number of times the next symbols is 0 at this state. Further, from [39, 13.2]

$$\begin{aligned} L(x^l|f) \geq & \sum_s n_s(x^l) H\left(\frac{n_{0|s}(x^l)}{n_s(x^l)}\right) - \frac{1}{2}\log n_s(x^l) \\ & - \frac{1}{2}\log\left(8\frac{n_{0|s}(x^l)}{n_s(x^l)}\frac{n_{1|s}(x^l)}{n_s(x^l)}\right) \\ & + \log(n_s(x^l) + 1) \end{aligned}$$

We want to upper bound the probability of the event $L(x^l|f) + \log l + \tau < l$. We can write

$$\log(n_s(x^l)+1) - \frac{1}{2}\log(n_s(x^l)) = \frac{1}{2}\log l + \log\left(\frac{1}{l}\frac{\left(n_s(x^l)+1\right)^2}{n_s(x^l)}\right)$$

let

$$r(x^l) = \sum_s n_s(x^l)\left(H\left(\frac{n_{0|s}(x^l)}{n_s(x^l)}\right) - 1\right)$$

and let $R(x^l)$ be the remaining small terms dependent on x^l,

$$\begin{aligned} R(x^l) = \sum_s -\frac{1}{2}\log\left(8\frac{n_{0|s}(x^l)}{n_s(x^l)}\frac{n_{1|s}(x^l)}{n_s(x^l)}\right) \\ + \log\left(\frac{1}{l}\frac{\left(n_s(x^l)+1\right)^2}{n_s(x^l)}\right). \end{aligned}$$

Then we have to upper bound (notice that $\sum_s n_s(x^l) = l$),

$$P\left(-r(x^l) - R(x^l) \geq \tau + \frac{k+2}{2}\log l\right)$$

The Chernoff bound is

$$\begin{aligned} &P\left(-r(x^l) - R(x^l) \geq \tau + \frac{k+2}{2}\log l\right) \\ &\leq \exp(-t(\tau + \frac{k+2}{2}\log l))M(t) \end{aligned}$$

or

$$\begin{aligned} &\ln P\left(-r(x^l) - R(x^l) \geq \tau + \frac{k+2}{2}\log l\right) \\ &\leq -t(\tau + \frac{k+2}{2}\log l) + \ln M(t) \end{aligned}$$

where

$$M(t) = E\left[\exp(-t(r(x^l) + R(x^l))\right]$$

In order to get a valid bound, we need to show that $M(t) < K < \infty$ independent of l for $t < \ln 2$. Now it's easy to see that $\exp(-tR(x^l)) \leq K < \infty$ for all t and l. So, we have to show

$$E\left[\exp(-t(r(x^l))\right] \leq K < \infty$$

We have to show that this is true for all state machines in the class of finite state machines with k states. Expanding this class does not reduce the maximum. So, instead consider the maximum over the following class of functions. A FSM with k states is a function $f(x^l) \in \{1, \ldots, k\}$ that satisfies that if $f(x^m) = f(\tilde{x}^m) = s$ then $f(x^m b) = f(\tilde{x}^m b)$. We extend the class by dispensing with the latter requirement. We can then describe the 'program' we run as follows. Based on x^m we choose a state $s_m \in \{1, \ldots, k\}$ *without* having any knowledge about x_{m+1}, except that it is independent and uniformly distributed. We can think of this slightly differently. The program puts x_{m+1} into one of k buckets, in order to maximize $E\left[\exp(-t(r(x^l))\right]$. It does so based on past data x^m. Now, as opposed to the state machine setup, the choice of s_m in no way restricts the choices of states (or buckets) s_n, $n > m$. Since the program has no knowledge of x_{m+1} the program cannot optimize s_m based on the *values* of x^m. Rather, it is sufficient to look at $n_s(m)$. It is now easy to see that the worst case is obtained if the bits are distributed evenly in the states, if l is a multiple of k. Thus, the worst case of $r(x^l)$ is

$$r(x^l) = \sum_s \frac{l}{k}\left(H\left(\frac{n_{0|s}(x^l)}{l/k}\right) - 1\right)$$

where the the $n_{0|s}(x^l)$ are independent over s. Thus, the problem is reduced to the case of a single state, which is showing that

$$E\left[\exp\left(tl\left(1 - H\left(\frac{k}{l}\right)\right)\right)\right] \leq K < \infty \tag{A.3.1}$$

Here we have

$$
\begin{aligned}
&E\left[\exp\left(tl\left(1-H\left(\frac{k}{l}\right)\right)\right)\right] \\
&=\sum_{k=0}^{l} 2^{\frac{t}{\ln 2}l(1-H(\frac{k}{l}))}\begin{pmatrix} k \\ l \end{pmatrix}2^{-l} \\
&=1+2\sum_{k=1}^{l/2} 2^{\frac{t}{\ln 2}l(1-H(\frac{k}{l}))}\begin{pmatrix} k \\ l \end{pmatrix}2^{-l} \\
&\leq 1+2\sum_{k=1}^{l/2} 2^{\frac{t}{\ln 2}l(1-H(\frac{k}{l}))}2^{-l}2^{lH(\frac{k}{l})}\sqrt{\frac{l}{\pi k(l-k)}} \\
&=1+2\sum_{k=1}^{l/2} 2^{\left(1-\frac{t}{\ln 2}\right)l(1-H(\frac{k}{l}))}\sqrt{\frac{l}{\pi k(l-k)}} \\
&\leq 1+2\sum_{k=1}^{l/2} 2^{-\left(1-\frac{t}{\ln 2}\right)\frac{2}{\ln 2}l\left(\frac{k}{l}-\frac{1}{2}\right)^2}\sqrt{\frac{l}{\pi k(l-k)}}
\end{aligned}
$$

where we have used [39, 13.2] and (A.1.4). The sum is actually decreasing as a function of l, but this seems hard to prove. Instead we upper bound the sum by

$$
\begin{aligned}
&\sum_{k=1}^{l/2} 2^{-\left(1-\frac{t}{\ln 2}\right)\frac{2}{\ln 2}l\left(\frac{k}{l}-\frac{1}{2}\right)^2}\sqrt{\frac{l}{\pi k(l-k)}} \\
&\leq \int_{1}^{l/2} 2^{-\left(1-\frac{t}{\ln 2}\right)\frac{2}{\ln 2}l\left(\frac{k}{l}-\frac{1}{2}\right)^2}\sqrt{\frac{l}{\pi k(l-k)}}dk
\end{aligned}
$$

Here we can upper bound $\sqrt{\frac{l}{\pi k(l-k)}} \leq \frac{\frac{4k\left(\sqrt{\frac{1}{l}}-1\right)}{k}+2}{\sqrt{\pi}}$ for $1 \leq k \leq \frac{l}{2}$. Then

$$\begin{aligned}
&\int_1^{l/2} 2^{-\left(1-\frac{t}{\ln 2}\right)\frac{2}{\ln 2} l\left(\frac{k}{l}-\frac{1}{2}\right)^2} \sqrt{\frac{l}{\pi k(l-k)}} dk \\
&\leq \int_{1/l}^{1/2} 2^{-\left(1-\frac{t}{\ln 2}\right)\frac{2}{\ln 2} l\left(x-\frac{1}{2}\right)^2} \frac{4x\left(\sqrt{\frac{1}{l}}-1\right)+2}{\sqrt{\pi}} l dx \\
&\leq \int_{-\infty}^{1/2} 2^{-\left(1-\frac{t}{\ln 2}\right)\frac{2}{\ln 2} l\left(x-\frac{1}{2}\right)^2} \frac{4x\left(\sqrt{\frac{1}{l}}-1\right)+2}{\sqrt{\pi}} l dx \\
&= \frac{K_1}{\sqrt{l}} + K_2
\end{aligned}$$

for some constants K_1, K_2, using Gaussian moments. This proves (A.3.1).

A.4 Linear Prediction

we showed

$$f(x^n|\tau,\mathbf{w}) = \frac{1}{(2\pi\tau)^{(n-M)/2}} \times \exp\left(-\frac{1}{2\tau}\left[\hat{r}_{(n)}(0) - 2\mathbf{w}^T\mathbf{p}_{(n)} + \mathbf{w}^T R^{(M)}_{(n)}\mathbf{w}\right]\right)$$

therefore using NLM we have

$$\begin{aligned} C(x^n) &= \int\int f(x^n|\tau,\mathbf{w})d\mathbf{w}d\tau \\ &= A\int_\tau \tau^{-\frac{(n-M)}{2}} \exp\left\{-\frac{\hat{r}_{(n)}(0)}{2\tau}\right\} e_1(\tau)\, d\tau \end{aligned}$$

where $A = \frac{1}{(2\pi)^{(n-M)/2}}$ and $e_1(\tau) = \int_{\mathbf{w}} \exp\left\{-\frac{1}{2\tau}\left[\mathbf{w}^T R_{(n)}\mathbf{w} - 2\mathbf{p}^T_{(n)}\mathbf{w}\right]\right\} d\mathbf{w}$. Hence

$$\begin{aligned} C(x^n) &= B\int_\tau \tau^{-\frac{n-2M}{2}} \exp\left\{-\frac{1}{2\tau}\left[\hat{r}_{(n)}(0) - \mathbf{p}^T_{(n)}R^{-1}_{(n)}\mathbf{p}_{(n)}\right]\right\} d\tau \\ &= B\int_\tau \tau^{-\frac{n-2M}{2}} \exp\left\{-\frac{1}{2\tau}\hat{\tau}^{(M)}_{(n)}\right\} d\tau \\ &= \frac{1}{2(\pi)^{\frac{n-2M}{2}}\sqrt{\det\left(R_{(n)}\right)}} \frac{\Gamma\left(\frac{n-2M-2}{2}\right)}{\left(\hat{\tau}^{(M)}_{(n)}\right)^{\frac{n-2M-2}{2}}} \end{aligned}$$

where $B = \frac{1}{(2\pi)^{(n-2M)/2}\sqrt{\det\left(R_{(n)}\right)}}$.

A.5 Vector Gaussian Case: unknown mean

we have to show the exponential term in right-hand side of equation (4.15) is equal to the exponential term in right-hand side of equation (4.14). Suppose

$$A = \frac{n}{(n+1)} \left(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}_n\right)^T \Sigma^{-1} \left(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}_n\right)$$
$$B = \left(\hat{\Sigma}_{n+1} - (n+1)\,\hat{\boldsymbol{\mu}}_{n+1}\hat{\boldsymbol{\mu}}_{n+1}^T - \hat{\Sigma}_n + n\hat{\boldsymbol{\mu}}_n\hat{\boldsymbol{\mu}}_n^T\right)\Sigma^{-1}$$

so we can write

$$\begin{aligned}
A &= \frac{n}{(n+1)} \left(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}_n\right)^T \Sigma^{-1} \left(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}_n\right) \\
&= \frac{n}{(n+1)}\mathbf{x}_{n+1}^T\Sigma^{-1}\mathbf{x}_{n+1} + \frac{n}{(n+1)}\hat{\boldsymbol{\mu}}_n^T\Sigma^{-1}\hat{\boldsymbol{\mu}}_n \\
&- \frac{2n}{(n+1)}\mathbf{x}_{n+1}^T\Sigma^{-1}\hat{\boldsymbol{\mu}}_n \\
&= \mathbf{x}_{n+1}^T\Sigma^{-1}\mathbf{x}_{n+1} - \frac{1}{n+1}\mathbf{x}_{n+1}^T\Sigma^{-1}\mathbf{x}_{n+1} \\
&+ n\hat{\boldsymbol{\mu}}_n^T\Sigma^{-1}\hat{\boldsymbol{\mu}}_n - \frac{n^2}{(n+1)}\hat{\boldsymbol{\mu}}_n^T\Sigma^{-1}\hat{\boldsymbol{\mu}}_n - \frac{2n}{(n+1)}\mathbf{x}_{n+1}^T\Sigma^{-1}\hat{\boldsymbol{\mu}}_n \\
&= \sum_{i=1}^{n+1}\mathbf{x}_i^T\Sigma^{-1}\mathbf{x}_i - \sum_{i=1}^{n}\mathbf{x}_i^T\Sigma^{-1}\mathbf{x}_i + n\hat{\boldsymbol{\mu}}_n^T\Sigma^{-1}\hat{\boldsymbol{\mu}}_n \\
&- \frac{1}{n+1}\left[\mathbf{x}_{n+1} + n\hat{\boldsymbol{\mu}}_n\right]^T\Sigma^{-1}\left[\mathbf{x}_{n+1} + n\hat{\boldsymbol{\mu}}_n\right] \\
&= \sum_{i=1}^{n+1}\mathbf{x}_i^T\Sigma^{-1}\mathbf{x}_i - (n+1)\,\hat{\boldsymbol{\mu}}_{n+1}^T\Sigma^{-1}\hat{\boldsymbol{\mu}}_{n+1} \\
&- \sum_{i=1}^{n}\mathbf{x}_i^T\Sigma^{-1}\mathbf{x}_i + n\hat{\boldsymbol{\mu}}_n^T\Sigma^{-1}\hat{\boldsymbol{\mu}}_n \\
&= B
\end{aligned}$$

A.6 Vector Gaussian Case: unknown variance

We showed that Σ has Inverse-Wishart distribution $\Sigma \sim \mathcal{W}_k^{-1}\left(\hat{\Sigma}_n, n\right)$ where $\hat{\Sigma}_n = \sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^T$, hence

$$f_{\mathbf{x}^n}(\Sigma) = \frac{\det\left(\hat{\Sigma}_n\right)^{\frac{n}{2}}}{2^{\frac{nk}{2}}\Gamma_k\left(\frac{n}{2}\right)} \det(\Sigma)^{-\frac{n+k+1}{2}} \operatorname{etr}\left\{-\frac{1}{2}\hat{\Sigma}_n\Sigma^{-1}\right\}$$

and since

$$f(\mathbf{x}_{n+1}|\Sigma) = \frac{1}{\sqrt{(2\pi)^k \det(\Sigma)}} \operatorname{etr}\left\{-\frac{1}{2}\left(\hat{\Sigma}_{n+1} - \hat{\Sigma}_n\right)\Sigma^{-1}\right\}$$

therefore we have

$$\begin{aligned}
f(\mathbf{x}_{n+1}|\mathbf{x}^n) &= \int_{\Sigma>0} f(\mathbf{x}_{n+1}|\Sigma)\, f_{\mathbf{x}^n}(\Sigma)\, d\Sigma \\
&= C \int_{\Sigma>0} \det(\Sigma)^{-\frac{n+k+2}{2}} \operatorname{etr}\left\{-\frac{1}{2}\hat{\Sigma}_{n+1}\Sigma^{-1}\right\} d\Sigma \\
&\overset{(A)}{=} C \int_{Y>0} \det(Y)^{\frac{n}{2}-\frac{k}{2}} \operatorname{etr}\left\{-\frac{1}{2}\hat{\Sigma}_{n+1}Y\right\} dY \\
&= D \int_{V>0} \det(V)^{\frac{n}{2}-\frac{k}{2}} \operatorname{etr}\{-V\}\, dV \\
&\overset{(B)}{=} D \int_{V>0} \det(V)^{\frac{n+1}{2}-\frac{k+1}{2}} \operatorname{etr}\{-V\}\, dV \\
&= \frac{1}{\pi^{\frac{k}{2}}} \frac{\det\left(\hat{\Sigma}_n\right)^{\frac{n}{2}}}{\det\left(\hat{\Sigma}_{n+1}\right)^{\frac{n+1}{2}}} \frac{\Gamma_k\left(\frac{n+1}{2}\right)}{\Gamma_k\left(\frac{n}{2}\right)}
\end{aligned}$$

where $C = \frac{\det\left(\hat{\Sigma}_n\right)^{\frac{n}{2}}}{2^{\frac{k(n+1)}{2}}\Gamma_k\left(\frac{n}{2}\right)\pi^{\frac{k}{2}}}$ and $D = \frac{\det\left(\hat{\Sigma}_n\right)^{\frac{n}{2}}}{\det\left(\hat{\Sigma}_{n+1}\right)^{\frac{n+1}{2}}}\frac{1}{\Gamma_k\left(\frac{n}{2}\right)\pi^{\frac{k}{2}}}$, and in equations (A) and (B) we changed the variable $\Sigma = Y^{-1}$ and $Y = 2\hat{\Sigma}_n^{-\frac{1}{2}}V\hat{\Sigma}_n^{-\frac{1}{2}}$ respectively and

$$\Gamma_m(a) = \int_{V>0} \det(V)^{a-\frac{(m+1)}{2}} \operatorname{etr}\{-V\}\, dV$$

is the multivariate Gamma function.

A.7 Vector Gaussian Case: unknown mean and variance

We showed that $\boldsymbol{\mu} \sim \mathcal{N}\left(\hat{\boldsymbol{\mu}}_n, \frac{1}{n}\Sigma\right)$ and $\Sigma \sim \mathcal{W}_k^{-1}\left(\hat{\Sigma}_n, n-1\right)$ where $\hat{\boldsymbol{\mu}}_n = \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i$ and $\hat{\boldsymbol{\Sigma}}_n = \sum_{i=1}^n \left(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_n\right)\left(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_n\right)^T$. Now using Bayes we can write the joint pdf as $f_{\mathbf{x}^n}\left(\boldsymbol{\mu}, \Sigma\right) = f_{\mathbf{x}^n}\left(\boldsymbol{\mu}|\Sigma\right) f_{\mathbf{x}^n}\left(\Sigma\right)$. Define

$$A \stackrel{\text{def}}{=} f\left(\mathbf{x}_{n+1}|\mathbf{x}^n\right) = \int_{\Sigma>0}\int f\left(\mathbf{x}_{n+1}|\boldsymbol{\mu}, \Sigma\right) f_{\mathbf{x}^n}\left(\boldsymbol{\mu}, \Sigma\right) d\boldsymbol{\mu} d\Sigma$$

so

$$A = B\int_{\Sigma>0} \det\left(\Sigma\right)^{-\frac{n+k+2}{2}} e_1\left(\Sigma\right) e_2\left(\Sigma\right) d\Sigma$$

where

$$\begin{aligned}
e_1\left(\Sigma\right) &= \operatorname{etr}\left\{-\frac{1}{2}\left(\hat{\Sigma}_n + n\hat{\boldsymbol{\mu}}_n\hat{\boldsymbol{\mu}}_n^T + \mathbf{x}_{n+1}\mathbf{x}_{n+1}^T\right)\Sigma^{-1}\right\} \\
e_2\left(\Sigma\right) &= \int \exp\left\{-\frac{n+1}{2}\left[\boldsymbol{\mu}^T\Sigma^{-1}\boldsymbol{\mu} - 2\hat{\boldsymbol{\mu}}_{n+1}\Sigma^{-1}\boldsymbol{\mu}\right]\right\} d\boldsymbol{\mu} \\
&= \sqrt{\frac{(2\pi)^k \det\left(\Sigma\right)}{(n+1)^k}} \exp\left\{\frac{n+1}{2}\hat{\boldsymbol{\mu}}_{n+1}^T\Sigma^{-1}\hat{\boldsymbol{\mu}}_{n+1}\right\} \\
B &= \frac{\det\left(\hat{\Sigma}_n\right)^{\frac{n-1}{2}}}{\Gamma_k\left(\frac{n-1}{2}\right)} \frac{n^{\frac{k}{2}}}{2^{\frac{k(n-1)}{2}}(2\pi)^k}
\end{aligned}$$

now since $\hat{\Sigma}_{n+1} = \hat{\Sigma}_n + n\hat{\boldsymbol{\mu}}_n\hat{\boldsymbol{\mu}}_n^T + \mathbf{x}_{n+1}\mathbf{x}_{n+1}^T - (n+1)\hat{\boldsymbol{\mu}}_{n+1}\hat{\boldsymbol{\mu}}_{n+1}^T$, by defining $C \stackrel{\text{def}}{=} B\sqrt{\frac{(2\pi)^k}{(n+1)^k}} =$

$\sqrt{\left(\frac{n}{n+1}\right)^k}\frac{\det\left(\hat{\Sigma}_n\right)^{\frac{n-1}{2}}}{\Gamma_k\left(\frac{n-1}{2}\right)}\frac{1}{2^{\frac{k(n-1)}{2}}(2\pi)^{\frac{k}{2}}}$ we can write

$$\begin{aligned}
A &= C\int_{\Sigma>0}\det(\Sigma)^{-\frac{n+k+1}{2}}\operatorname{etr}\left\{-\frac{1}{2}\hat{\Sigma}_{n+1}\Sigma^{-1}\right\}d\Sigma \\
&= C\int_{Y>0}\det(Y)^{\frac{n}{2}-\frac{k+1}{2}}\operatorname{etr}\left\{-\frac{1}{2}\hat{\Sigma}_{n+1}Y\right\}dY \\
&= C\frac{2^{\frac{kn}{2}}}{\det\left(\hat{\Sigma}_{n+1}\right)^{\frac{n}{2}}}\int_{V>0}\det(V)^{\frac{n}{2}-\frac{k+1}{2}}\operatorname{etr}\{-V\}dV \\
&= \frac{1}{\pi^{\frac{k}{2}}}\sqrt{\left(\frac{n}{n+1}\right)^k}\frac{\det\left(\hat{\Sigma}_n\right)^{\frac{n-1}{2}}}{\det\left(\hat{\Sigma}_{n+1}\right)^{\frac{n}{2}}}\frac{\Gamma_k\left(\frac{n}{2}\right)}{\Gamma_k\left(\frac{n-1}{2}\right)}
\end{aligned}$$